NONLINEAR MODEL REDUCTION OF STOCHASTIC MICRODYNAMICS

Wyatt H. Bridgman

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Mathematics.

Chapel Hill
2021

Approved by:

Sorin Mitran

Jeremy Marzuola

Katie Newhall

Jingfang Huang

Boyce Griffith

ABSTRACT

Wyatt H. Bridgman: Nonlinear Model Reduction of Stochastic Microdynamics
(Under the direction of Sorin Mitran)

This thesis presents a nonlinear model reduction procedure for stochastic microdynamics models that possess mesoscale separation between fast and slow dynamics. Model reduction procedures typically reduce the dimension of deterministic dynamical systems through linear projection operators which offer limited compression capabilities for nonlinear systems. On the other hand, deep neural networks provide a class of nonlinear transformations for regression that can approximate arbitrarily complex functions. The approach developed in this thesis attempts to carry out nonlinear model reduction of stochastic models using deep neural networks to approximate a transformation onto reduced coordinates taken to be the parameters of the network. The stochasticity of the microdynamics is inherited by the reduced, mesoscale model by viewing the parameters as stochastic processes. Moderate time scale separation suggests that non-Gaussian behavior must be considered in contrast with the convergence to Gaussian noise in the limit of infinite timescale separation provided by homogenization theory. This thesis considers several approaches for modeling the stochastic processes concluding with an information geometric strategy for estimating probability distribution functions.

The procedure is applied to protein folding within molecular dynamics simulations, a widely used technique to model large collections of atoms which interact through nonlinear forces and are driven by a stochastic heat bath. Protein folding occurs on a larger, mesoscale with respect to the timescale of numerical integration.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AE	Autoencoder
DNN	Deep Neural Network
DNS	Direct Numerical Simulation
FFNN	Feedforward Neural Network
MD	Molecular Dynamics
PCA	Principal Component Analysis
POD	Proper Orthogonal Decomposition
SVD	Singular Value Decomposition

CHAPTER 1: INTRODUCTION

Nonlinear systems with a large number of degrees of freedom arise frequently in science and engineering problems. To model unresolved physics, these systems may also include stochastic components so that the evolution of the system becomes a stochastic process with complex statistical properties. It is typically the behavior of large components of these systems over meso or macro time scales that is of particular relevance to the application. Often, only micro scale physical laws are explicitly available forcing the use of computationally expensive micro-scale simulations. Consequently, model reduction procedures that can accelerate computation are of great interest in a wide range of fields. This thesis develops a general nonlinear model reduction procedure for stochastic dynamcal systems using deep neural networks as nonlinear transformations into lower-dimensional spaces. A particular focus of this thesis is modeling the weights of neural networks as stochastic processes and the use of methods from information geometry [6] to extract their statistical properties. The thesis will consider molecular dynamics of a simple protein as a particular application for the model reduction procedure. Before describing this acceleration procedure, it will be beneficial to discuss the basic theory of model reduction [20], machine learning [60], and molecular dynamics [77] . This will provide context that motivates the approach comprising the central topic of this thesis.

1.1: MODEL REDUCTION

First, it is necessary to define the concept of model reduction. Given a dynamical system defined on a high-dimensional phase space $\mathcal{X}$, the assumption of model reduction is that the dynamics of interest approximately take place in some subspace $\mathcal{Y} \subset \mathcal{X}$. The objective is then to find a self-contained description of these dynamics on the lower-dimensional subspace $\mathcal{Y}$ without fully resolving the dynamics in $\mathcal{X} \setminus \mathcal{Y}$. This description is obtained through a mathematical procedure which transforms the full, high-dimensional model into a low-dimensional model that best captures the behavior of the full system according to some metric. Model reduction is closely related to dimensionality reduction [47, 19, 95], which concerns the extraction of

a a lower-dimensional description of high-dimensional data, but requires additional treatment of the features defining a dynamical system such as vector fields.

As this thesis focuses on reducing dimension of a dynamical system, it will be useful to discuss the general structure of a model reduction procedure. Many engineering problems can be modeled as a nonlinear partial differential equation governing some underlying physical field. Discretization of a scalar PDE in one spatial variable results in a system of ODEs of the form

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \boldsymbol{V}_{\mathcal{X}}(\mathbf{x})$$

where $\mathbf{x} \in \mathbb{R}^D$ is a large-dimesional vector of state variables and $\boldsymbol{V}_{\mathcal{X}} : \mathbb{R}^D \to \mathbb{R}^D$ a possibly nonlinear function defining a vector field on the phase space $\mathcal{X} \subset \mathbb{R}^D$. Model reduction seeks a lower-dimensional manifold $\mathcal{Y} \subset \mathcal{X}$ of reduced variables on which the phase space trajectories approximately reside. This is also called an invariant manifold [43] for the dynamical system when phase space trajectories strictly remain within the manifold as the forward propagation operator for any time $t \in \mathbb{R}_{\geqslant 0}$ then maps the manifold to itself. This low-dimensional manifold for the dynamics can be described by a function $\boldsymbol{F} : Y \to \mathcal{X}$ where $Y \subset \mathbb{R}^d$, $d \ll D$, is the space of reduced variables. Given Y, the reduced phase space of states, it remains to construct a vector field $\boldsymbol{V}_{\mathcal{Y}}$ on Y specifying the dynamics of the reduced model. This is done in a series of steps: computing the value of $\boldsymbol{V}_{\mathcal{X}}$ at $\boldsymbol{F}(\mathbf{y}) \in \mathcal{Y}$, projecting this vector onto the tangent space $T_{\boldsymbol{F}(\mathbf{y})}\mathcal{Y}$, and finally pulling this vector back to Y. This induces a differential equation on Y of the form

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \boldsymbol{V}_{\mathcal{Y}}(\mathbf{y}) = (D_{\mathbf{y}}\boldsymbol{F})^{-1}\mathbf{P}\boldsymbol{V}_{\mathcal{X}}(\boldsymbol{F}(\mathbf{y})) \tag{1.1}$$

where $\mathbf{y} \in Y$ and $\mathbf{P} = \mathbf{P}_{T_{\boldsymbol{F}(\mathbf{y})}\mathcal{Y}}$ is projection onto the tangent space of $\mathcal{Y}$ at $\boldsymbol{F}(\mathbf{y})$.

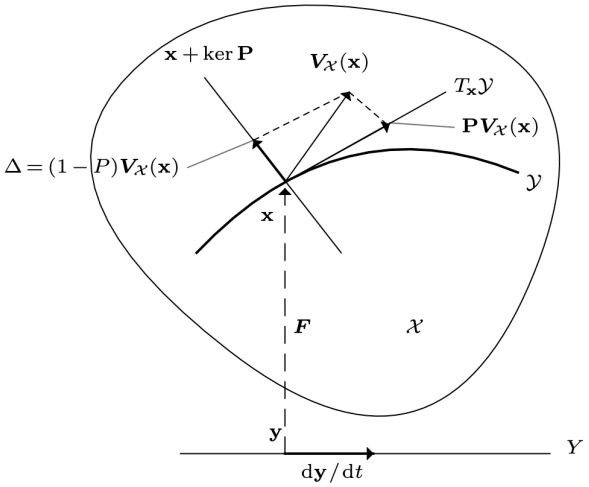


Figure 1.1: Induced dynamics on reduced space W

If $\mathcal{Y}$ is truly an invariant manifold, we would have that

$$\Delta(\mathbf{x}) = (\mathbf{I} - \mathbf{P})\boldsymbol{V}_{\mathcal{X}}(\mathbf{x}) = \mathbf{0} \tag{1.2}$$

for all $\mathbf{x} \in \mathcal{Y}$, i.e., the defect of invariance Δ vanishes over $\mathcal{Y}$. In practice, Δ is minimized to obtain a reduced model. The linear subspace $\mathbf{x} + \ker(P)$ defines the space of fast dynamics that is not resolved in the reduced model [43].

1.1.1 Deterministic model reduction

In this section, we consider model reduction approaches in the case where the dynamics on $\mathcal{X} \setminus \mathcal{Y}$ are not resolved at all. Hence, $\Delta(\mathbf{x})$ is implicitly assumed to be small on the low-dimensional manifold and the effect of the neglected modes of the system are not considered. In subsequent sections, stochastic model reduction approaches are discussed which represent the neglected modes as stochastic terms yielding a system of SDEs as the reduced system.

1.1.1.1 Model reduction with linear subspaces

There are a number of well-established mathematical tools [20] for constructing linear subspaces that capture dominant modes of functions and data. Hence, the most widely used model reduction techniques seek linear subspaces on which to project the dynamical system. The general structure of this approach can

be described by first considering a generic dynamical system of the form

$$\begin{aligned} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= \mathbf{A}\mathbf{x} + \boldsymbol{f}(\mathbf{x}) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{x}(0) &= \mathbf{x}_0 \end{aligned} \tag{1.3}$$

$$\mathbf{x} \in \mathbb{R}^D, \mathbf{A}, \mathbf{B} \in \mathbb{R}^{D\times D}, \mathbf{u}(t) : \mathbb{R} \to \mathbb{R}^D$$

where $\mathbf{A}$ and $\boldsymbol{f}$ represent the linear and nonlinear components of the dynamics, respectively, and $\boldsymbol{u}(t)$ the input to the system or forcing term. Model reduction then seeks linear subspaces $\mathcal{V}, \mathcal{W} \subset \mathbb{R}^D, \dim(\mathcal{V}) = \dim(\mathcal{W}) = d$ defined by matrices $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{D\times d}$ in the sense that

$$\mathrm{range}(\mathbf{V}) = \mathcal{V}, \mathrm{range}(\mathbf{W}) = \mathcal{W}$$

The ansatz $\mathbf{x}(t) \approx \mathbf{V}\mathbf{y}(t), \mathbf{y}(t) \in \mathbb{R}^d$ is made which, when substituted into ()1.3), yields the differential equation

$$\mathbf{V}\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{A}\mathbf{V}\mathbf{y}(t) + \boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) + \mathbf{B}\mathbf{u}(t) + \mathbf{r}(t) \tag{1.4}$$

where the residual $\mathbf{r}(t) \in \mathbb{R}^D$ accounts for the fact that $\mathbf{V}\mathbf{y}(t)$ is not necessarily an exact solution of equation 1.3. To obtain a reduced equation defined on $\mathbb{R}^d, d \ll D$, we must specify which vector the right-hand side of 1.4 refers to. This amounts to a choice of residual $\mathbf{r}(t)$. As $\mathbf{V}\mathbf{y}(t)$ is constrained to $\mathcal{V}$, consistency requires that this vector is also an element of $\mathcal{V}$. Since we would like to capture the behavior of the full system in the reduced model, it makes sense to choose the vector in $\mathcal{V}$ closest to $\mathbf{A}\mathbf{V}\mathbf{y}(t) + \boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) + \mathbf{B}\mathbf{u}(t)$ with respect to some norm picked appropriately for the problem. A common choice is the $\mathcal{L}_2$ norm which is equivalent to minimizing the difference in energy between the full and reduced models. Minimizing the $\mathcal{L}_2$ norm amounts to requiring the residual to be orthogonal to the subspace $\mathcal{W}$ by setting

$$\mathbf{W}^T\mathbf{r}(t) = \mathbf{0}$$

Left-multiplying by $\mathbf{W}^T$ gives the Petrov-Galerkin projection reduced order model as

$$\begin{aligned} \mathbf{W}^T\mathbf{V}\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= \mathbf{W}^T\mathbf{A}\mathbf{V}\mathbf{y}(t) + \mathbf{W}^T\boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) + \mathbf{W}^T\mathbf{B}\mathbf{u}(t) \\ &\Updownarrow \\ \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= (\mathbf{W}^T\mathbf{V})^{-1}\mathbf{W}^T[\mathbf{A}\mathbf{V}\mathbf{y}(t) + \boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) + \mathbf{B}\mathbf{u}(t)] \end{aligned}$$

which can be considered a case of oblique-projection when lifted back into the full-dimesional space $\mathbb{R}^D$. This is seen through left multiplication by $\mathbf{V}$

$$\mathbf{V}\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{V}(\mathbf{W}^T\mathbf{V})^{-1}\mathbf{W}^T[\mathbf{A}\mathbf{V}\mathbf{y}(t) + \boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) + \mathbf{B}\mathbf{u}(t)]$$

and then making the identification $\tilde{\mathbf{x}}(t) = \mathbf{V}\mathbf{y}$ to obtained the lifted reduced model

$$\frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}t} = \underbrace{\mathbf{V}(\mathbf{W}^T\mathbf{V})^{-1}\mathbf{W}^T}_{\text{oblique projection}}[\mathbf{A}\tilde{\mathbf{x}}(t) + \boldsymbol{f}(\tilde{\mathbf{x}}(t)) + \mathbf{B}\mathbf{u}(t)]$$

where $\mathbf{V}(\mathbf{W}^T\mathbf{V})^{-1}\mathbf{W}^T$ represents oblique projection onto $\mathcal{V}$ parallel to $\mathcal{W}$.

The case where $\mathbf{W} = \mathbf{V}$ constitutes the Galerkin projection method, which when $\mathbf{V}$ is orthogonal, takes the form

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{V}^T[\mathbf{A}\mathbf{V}\mathbf{y}(t) + \boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) + \mathbf{B}\mathbf{u}(t)]$$

When $\mathbf{u}(t) = \mathbf{0}$, this becomes

$$\begin{aligned} \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= \mathbf{V}^T[\mathbf{A}\mathbf{V}\mathbf{y}(t) + \boldsymbol{f}(\mathbf{V}\mathbf{y}(t))] \\ &\Updownarrow \\ \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= (D_{\mathbf{y}}\boldsymbol{F})^{-1}\mathbf{P}\boldsymbol{V}_{\mathcal{X}}(\boldsymbol{F}(\mathbf{y})) \end{aligned}$$

which can be seen as a special case of 1.1 where

$$\begin{aligned} \boldsymbol{F}(\mathbf{y}) &= \mathbf{V}\mathbf{y} \\ \boldsymbol{V}_{\mathcal{X}}(\mathbf{x}) &= \mathbf{A}\mathbf{x} + \boldsymbol{f}(\mathbf{x}) \\ P &= \mathbf{V}\mathbf{V}^T \\ (D_{\mathbf{y}}\boldsymbol{F})^{-1} &= \mathbf{V}^T \end{aligned}$$

This provides a general framework for obtaining a reduced model given linear subspaces $\mathbf{V}, \mathbf{W}$.

1.1.1.2 Basis construction methods

The previous section assumed we had already selected appropriate linear subspaces through matrices $\mathbf{V}, \mathbf{W}$ but did not discuss how to obtain them. This subsection presents three different methods for deriving the reduced basis matrices: proper orthogonal decomposition (POD), rational interpolation, and balanced truncation.

Proper Orthogonal Decomposition One of the most widely applicable basis construction methods is proper orthogonal decomposition (POD) introduced orginally in [64] to study turbulent fluid flows. It is closely related to methods from other fields such as Karhunen-Loeve expansions of stochastic processes [63, 56] and principal component analysis in statistics and data science [48, 51]. POD efficiency is a based on the snapshot approach [92] which relies on samples from phase space trajectories of a dynamical system. This point cloud of samples is used to exact a linear subspace of dominant modes of the system that is optimal in the least squares sense.

Consider s snapshots $\mathbf{x}_1, \ldots, \mathbf{x}_{n_s} = \mathbf{x}(t_1), \ldots, \mathbf{x}(t_{n_s})$ of the system (1.3) at times $t_1, \ldots, t_{n_s}$ and collect them into a data matrix

$$\mathbf{X} = [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_{n_s}]$$

The singular value decomposition of $\mathbf{X} \in \mathbb{R}^{D \times n_s}$ is written as

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T$$

where $\mathbf{U} \in \mathbb{R}^{D\times D}, \mathbf{\Sigma} \in \mathbb{R}^{D\times n_s}, \mathbf{Y} \in \mathbb{R}^{n_s\times n_s}$. The matrices $\mathbf{U}$, $\mathbf{Y}$ form orthonormal bases for $\mathbb{R}^D$ and R^{n_s}, respectively and $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_k)$ where $\sigma_1 \geqslant \sigma_2 \cdots \geqslant \sigma_k > 0$ are the singular values. The POD reduced basis $\mathbf{U}_r$ is chosen as the first r columns of $\mathbf{U}$ corresponding to the largest r singular values. The POD basis $\mathbf{U}_r$ is orthonormal and is optimal in the sense that

$$\min_{\mathbf{U}_r\in\mathbb{R}^{D\times r}} \|\mathbf{X} - \mathbf{U}_r\mathbf{U}_r^T\mathbf{X}\|_F^2 = \min_{\mathbf{U}_r\in\mathbb{R}^{D\times r}} \sum_{i=1}^{n_s} \|\mathbf{x}_i - \mathbf{U}_r\mathbf{U}_r^T\mathbf{x}_i\|_2^2 = \sum_{i=r+1}^{n_s} \sigma_i^2$$

The squared reconstruction error is given by the sum of the singular values of the neglected basis vectors and hence the distribution of singular values can guide the appropriate choice of dimension for the reduced basis.

POD can also be carried out in the frequency domain for linear systems [55]. Through an application of Parseval's theorem the POD kernel $\mathbf{K}_t = \mathbf{X}\mathbf{X}^T$ has the frequency domain representation

$$\mathbf{K}_\omega = \frac{1}{2\pi}\sum_{i=1}^{n_s} \bar{\mathbf{x}}_i\bar{\mathbf{x}}_i^* \Delta\omega_i$$

where $\bar{\mathbf{x}}_i$ is the ith frequency domain snapshot. This approach to POD has been used to create a procedure to approximate balanced trunction in [99].

Rational Interpolation Rational interpolation [16, 11] is a basis construction technique for linear, time-invariant (LTI) systems of the form

$$\begin{aligned} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) \\ \mathbf{x}(0) &= \mathbf{x}_0 \end{aligned} \tag{1.5}$$

where $\mathbf{x} \in \mathbb{R}^D$ is a vector of state variables, $\mathbf{u} \in \mathbb{R}^p$ a vector of input variables, and $\mathbf{y} \in \mathbb{R}^q$ a vector output variables related to the state through the linear transformation $\mathbf{C} \in \mathbb{R}^{q\times D}$. Typically it is the case that $p, q \ll D$. The output $\mathbf{y}(t)$ of an LTI is given by the convolution of the input $\mathbf{u}(t)$ with the systems impulse response $\mathbf{h}(t)$. In the one-dimensional case, this takes the form

$$y(t) = \int_{-\infty}^{\infty} u(t-\tau)h(\tau)\,\mathrm{d}\tau$$

where the impulse response $h(t)$ defines the output of the system in response to a delta function input. The Laplace transform of the impulse response in the general case then provides the transfer function $\mathbf{H}(s)$. Convolution with $\mathbf{h}(t)$ with respect to time becomes simple multiplication with $\mathbf{H}(s)$ in frequency space so that $\mathbf{H}(s)$ provides the frequency space characterization of the LTI system's input-ouput mapping.

Rational interpolation is a strategy which constructs basis matrices $\mathbf{V}$ and $\mathbf{W}$ by requiring that the transfer function of the reduced model interpolates the transfer function of the full model in some manner. One approach is moment-matching [14, 39] where the reduced basis is determined that satisfies the Hermite interpolation conditions

$$\frac{d^k}{ds^k}\boldsymbol{H}(\hat{s}) = \frac{d^k}{ds^k}\boldsymbol{H}_r(\hat{s}); k = 0, 1, \ldots, N$$

Moments can also be matched around multiple expansion points leading to multipoint moment matching [10, 11]. This can be extended to the parametric case using multivariate Taylor expansion of $\boldsymbol{H}(s, \boldsymbol{p})$ about $(\hat{s}, \boldsymbol{p})$ [21, 45].

Balanced Truncation Balanced truncation [68, 66] is a systems theory approach to model reduction for linear time-invariant (LTI) systems of the form 1.5. The balanced truncation basis matrices $\mathbf{V}$ and $\mathbf{W}$ depend on the two system Gramians defined by

$$\begin{aligned}\boldsymbol{\mathcal{P}} &= \int_0^\infty e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^* e^{\mathbf{A}^* t}\mathrm{d}t \\ \boldsymbol{\mathcal{Q}} &= \int_0^\infty e^{\mathbf{A}t}\mathbf{C}^*\mathbf{C} e^{\mathbf{A}t}\mathrm{d}t\end{aligned}$$

which represent the reachability and observability Gramian, respectively. The reachability of a state $\boldsymbol{x}$ is a measure of how easy it is to reach $\boldsymbol{x}$ from the zero state. The observability of a state $\boldsymbol{x}_0$ is a measure of how easy it is to distinguish the initial state $\boldsymbol{x}_0$ from the zero state by observing the output $\boldsymbol{y}(t)$ with zero input $\boldsymbol{u}(t) = \mathbf{0}$. These energies are defined by the two Gramians $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{Q}}$. Model reduction is achieved by eliminating portions of the state space which are less important with respect to these measures.

Balanced truncation has been extended to nonlinear systems [86] but is currently infeasible for even moderately large systems.

1.1.1.3 Parametric dynamical systems

Many engineering problems have physics which depend on a collection of parameters $\mathbf{p} \in \Omega \subset \mathbb{R}^p$. Discretizations of these problems yield dynamical systems of the form

$$\begin{aligned} \mathbf{E}(\mathbf{p})\frac{\mathrm{d}\mathbf{x}(t;\mathbf{p})}{\mathrm{d}t} &= \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \mathbf{B}(\mathbf{p})\mathbf{u}(t) \\ \mathbf{y}(t;\mathbf{p}) &= \boldsymbol{C}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) \end{aligned}$$

where $\mathbf{y}(t;\mathbf{p}) \in \mathbb{R}^q$, $\mathbf{u}(t) \in \mathbb{R}^m$, $\mathbf{x}(t;\mathbf{p}) \in \mathbb{R}^D$ and $\mathbf{C}(\mathbf{p}) \in \mathbb{R}^{q\times D}$ defines the observables $\mathbf{y}$ with respect to the system state $\mathbf{x}$. Note that the matrix operators now posses parametric dependence on $\mathbf{p}$ which may be nonlinear. Standard projection-based methods like POD yield a reduced model

$$\begin{aligned} \mathbf{E}_r(\mathbf{p})\frac{\mathrm{d}\mathbf{x}_r(t;\mathbf{p})}{\mathrm{d}t} &= \mathbf{A}_r(\mathbf{p})\mathbf{x}_r(t;\mathbf{p}) + \mathbf{B}_r(\mathbf{p})\mathbf{u}(t) \\ \mathbf{y}_r(t;\mathbf{p}) &= \boldsymbol{C}_r(\mathbf{p})\mathbf{x}_r(t;\mathbf{p}) \end{aligned}$$

where $\mathbf{x}_r(t;\mathbf{p}) \in \mathbb{R}^d$ is the reduced state and reduced parametric matrix operators have the form $\mathbf{A}_r(\mathbf{p}) = \mathbf{W}(\mathbf{p})^T\mathbf{A}(\mathbf{p})\mathbf{V}(\mathbf{p})$ for some bases $\mathbf{V}(\mathbf{p}), \mathbf{W}(\mathbf{p}) \in \mathbb{R}^{D\times d}$ [22]. The goal is to capture the parametric dependence of the reduced matrices without having to recompute $\mathbf{V}(\mathbf{p}), \mathbf{W}(\mathbf{p})$ and $\mathbf{A}_r(\mathbf{p}) = \mathbf{W}(\mathbf{p})^T\mathbf{A}(\mathbf{p})\mathbf{V}(\mathbf{p})$ for every parameter value. One approach is to use global basis matrices $\mathbf{V}, \mathbf{W}$ determined by sampling many parameter values $\mathbf{p}_1, \ldots, \mathbf{p}_{n_s}$ which capture the global variation of the reduced basis.

Interpolation across local data Given local basis matrices, a basis matrix for a new parameter value can be generated through interpolation. Naive interpolation between bases can yield matrices which do not form a basis or have certain necessary properties such as orthogonality. One approach to mitigating this issues is to interpolate the underlying subspace rather than working on the basis vectors themselves [8]. This is based on the concept of a Grassmann manifold [1] $\mathbf{Gr}_k(V)$, a smooth manifold parametrizing the set of possible k-dimensional linear subpsaces of an$n$-dimensional vector space V. The strategy is to map subspaces to the tangent space of a Grassmann manifold, carry out interpolation in the tangent space where the interpolation process is straightforward and then map back to the manifold. Interpolation on matrix manifolds is another approach [103, 7, 9] which allows the interpolation process to preserve constraints such as orthogonality.

Interpolating the local basis matrices has the disadvantage that when the new basis matrices are computed for a given parameter value $\mathbf{p}$, the product $\mathbf{A}_r(\mathbf{p}) = \mathbf{W}(\mathbf{p})^T\mathbf{A}(\mathbf{p})\mathbf{V}(\mathbf{p})$ must be recomputed which depend on the original system dimension D and are thus expensive. In [9, 74] a congruence transformation is first performed for local basis matrices so that the reduced systems are expressed in the same generalized coordinate system. Matrices can then be interpolated using matrix manifold interpolation [103, 7, 9] and direct interpolation [74].

Another option is to interpolate local transfer functions of local reduced models [18]. If we have sampled parameter points $\boldsymbol{p}_k, k = 1, \ldots, K$, the local reduced models have transfer functions

$$\boldsymbol{H}_r(s, \boldsymbol{p}_k) = \boldsymbol{C}_r(\boldsymbol{p}_k)(s\boldsymbol{E}_r(\boldsymbol{p}_k) - \boldsymbol{A}_r(\boldsymbol{p}_k))^{-1}\boldsymbol{B}_r(\boldsymbol{p}_k)$$

The reduced order transfer function at a new paramter value can be obtained by interpolation using a Lagrange basis functions on the samples points [17]. Each local transfer function is independent of the representation of its local basis so nonuniqueness is not an issue as it is for state-space representations.

1.1.1.4 Linear subspace approach for nonlinear systems

Because this thesis is concerned with model reduction for nonlinear dynamical sytems, it will be useful to look specifically at the linear subspace approach in the case of a nonlinear system and the issues that arise. The general form of a nonlinear system is given by

$$\begin{aligned} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= \boldsymbol{f}(\mathbf{x}, t) \\ \mathbf{x}(0) &= \mathbf{x}_0 \end{aligned} \tag{1.6}$$

where again $\mathbf{x} \in \mathbb{R}^D$ and $\boldsymbol{f}(\mathbf{x}, t) : \mathbb{R}^D \times \mathbb{R} \to \mathbb{R}^D$ provides the nonlinearity. Given a $d \ll D$ dimensional POD subspace $\mathcal{V}$ defined by an orthonormal matrix $\mathbf{V} \in \mathbb{R}^{D\times d}$, the reduced and projected systems defined

by state vectors $\mathbf{y} \in \mathbb{R}^d, \tilde{\mathbf{x}} \in \mathbb{R}^D$, respectively, are governed by the equations

$$\begin{aligned} \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= \mathbf{V}^T \boldsymbol{f}(\mathbf{V}\mathbf{y}, t) \\ \mathbf{y}(0) &= \mathbf{V}^T \mathbf{x}_0 \end{aligned}$$

$$\begin{aligned} \frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}t} &= \mathbf{V}\mathbf{V}^T \boldsymbol{f}(\tilde{\mathbf{x}}, t) \\ \tilde{\mathbf{x}}(0) &= \mathbf{V}\mathbf{V}^T \mathbf{x}_0 \end{aligned}$$

An estimate for the error $\mathbf{e}(t) = \tilde{\mathbf{x}}(t) - \mathbf{x}(t)$ is derived in [81]. Denote the component of $\mathbf{e}(t)$ orthogonal to the subspace $\mathcal{V}$ by $\mathbf{e}_0(t)$ and the parallel component by $\mathbf{e}_i(t)$. The error component $\mathbf{e}_0(t)$ comes from the subspace approximation and represents the error between $\mathbf{x}(t)$ and its projection onto $\mathcal{V}$ denoted $\hat{\mathbf{x}}(t)$. Because we are also projecting the vector-field onto $\mathcal{V}$ we make further approximations resulting in an additional error $\mathbf{e}_i(t)$.

To understand the error, an initial value problem for $e_i(t)$ of the form

$$\dot{\mathbf{e}}_i = \mathbf{P}(\boldsymbol{f}(\mathbf{x}(t) + \mathbf{e}_0(t) + \mathbf{e}_i, t) - \boldsymbol{f}(\mathbf{x}(t), t)); \mathbf{e}_i(0) = 0$$

is considered where $\mathbf{P}$ is the projection operator onto the reduced subspace $\mathcal{V}$. Error bounds for e_i can be proven in the linear case and also in the general nonlinear case. In the linear case, it can be understood how the rate of change of the components of the vector field parallel to $\mathcal{V}$ in the directions orthogonal to $\mathcal{V}$ affect the $\mathbf{e}_i$ component of the error. In the extreme case where the components of the vector field parallel to $\mathcal{V}$ are invariant in directions perpendicular to $\mathcal{V}$, we have that $\mathbf{e}_i$ is zero. This corresponds to $\mathbf{A}$ being a block normal matrix in the linear case where $\boldsymbol{f}(\mathbf{x}) = \mathbf{A}\mathbf{x}$. Furthermore, if $\mathbf{A}$ is symmetric, then Galerkin projection preserves the stability of the full-dimensional model in the reduced model.

The approximation error introduced by projection of the state and vector field onto $\mathcal{V}$ is not the only source of issues. Another problem is the computational complexity of evaluating terms of the form $\mathbf{V}^T \boldsymbol{f}(\mathbf{V}\mathbf{y}, t)$, that when $\boldsymbol{f}$ has no special structure, require $\mathcal{O}(D)$ operations to compute.

When the full model contains only polynomial nonlinearities in the state, the POD approach preserves this structure and can be efficiently evaluated with recourse to high-dimensional quantities [79]. For example,

if only quadratic nonlinearities are present, the ODE system can be written as

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{A}\mathbf{x} + \mathbf{F}(\mathbf{x} \otimes \mathbf{x})$$

where $\mathbf{A} \in \mathbb{R}^{D \times D}$ is the linear part of the vector field and $\mathbf{F} \in \mathbb{R}^{D \times D^2}$ a matricized tensor operator. The POD reduced model is then given by

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \hat{\mathbf{A}}\mathbf{y} + \hat{\mathbf{F}}(\mathbf{y} \otimes \mathbf{y})$$

where

$$\hat{\mathbf{A}} = \mathbf{V}^T\mathbf{A}\mathbf{V}, \widehat{\mathbf{F}} = \mathbf{V}^{\boldsymbol{T}}\mathbf{F}(\mathbf{V} \otimes \mathbf{V})$$

are reduced matrix operators. The cost of evaluating the reduced model depends only on the reduced dimension d.

To decrease the computational complexity when $\boldsymbol{f}$ does not have special structure, Taylor series expansions of nonlinear terms can be used [26, 44, 96] as well as selective sampling of nonlinear terms combined with interpolation. The latter approach is taken by a number of methods including: missing point estimation [13], Gauss-Newton with approximated tensors [27], the gappy POD interpolation method [38], the empirical interpolation method [15, 37], and discrete empirical interpolation method (DEIM) [30].

For example, the strategy of the DEIM approach is to approximate the nonlinear function by projecting it onto a subspace that approximates the space generated by the nonlinear function and that is spanned by a basis of dimension $d \ll D$. Hence, $\boldsymbol{f}(\mathbf{V}\mathbf{y}(t)) = \boldsymbol{g}(t)$ is approximated using the POD basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d\} \subset \mathbb{R}^d$ in the form

$$\boldsymbol{g}(\tau) \approx \mathbf{V}\boldsymbol{c}(t)$$

where $\boldsymbol{c}(t)$ is a coefficient vector to be determined.

1.1.1.5 Model reduction with nonlinear submanifolds

In the previous sections the model reduction procedures sought invariant manifolds for reduced dynamics in the form of linear subspaces. Given a nonlinear dynamical system of the form 1.6, there is no reason to suspect that an invariant manifold of system would take the form of a linear space. In general, invariant subspaces could be nonlinear manifolds Ω immersed in the phase space $U \subset \mathbb{R}^D$ possessing curvature and topological features. Constructing a nonlinear manifold immersed in a high-dimensional space $\mathbb{R}^D$ is more

difficult problem. For example, projection operators mapping the high dimensional vector field $\boldsymbol{V}_{\mathcal{X}}(\mathbf{x})$ onto the tangent spaces $T_p\mathcal{Y}$ of $\mathcal{Y}$ become point-dependent fields. Also, manifolds in general may require multiple coordinate functions defined on overlapping neighborhoods to define coordinates on the entire structure. In the case where the manifold is defined by a single coordinate patch $\boldsymbol{F} : Y \to \mathcal{X}$ where $Y \subset \mathbb{R}^d$, $d \ll D$, a well-developed approach exists for constructing or approximating the invariant manifold. This restricts invariant manifolds to spaces diffeomorphic to some $\mathbb{R}^d$ but is instructive in describing model reduction in its most general form.

The invariance equation 1.2, which can also be written in the form

$$\Delta_{\mathbf{y}} = (1 - \mathbf{P}_{T_{\boldsymbol{F}(\mathbf{y})}})\boldsymbol{V}_{\mathcal{X}}(\boldsymbol{F}(\mathbf{y})) = \mathbf{0}$$

is a differential equation for the unknown map $\boldsymbol{F} : Y \to \mathcal{X}$. Hence, the Newton method can be used as an iterative method for solving it. For each iteration, the invariance equation is linearized and then solved. A first order approximation for $\boldsymbol{V}_{\mathcal{X}}(\boldsymbol{F}(\mathbf{y}))$ and zero order approximation for $\mathbf{P}_{T_{\boldsymbol{F}(\mathbf{y})}}$ are used. This method leads to the slowest invariant manifold [42].

Another approach for constructing the invariant manifold is to consider manifolds immersed in phase space and study their evolution along dynamical system trajectories. The motion of the manifold along itself is subtracted leaving only the component of the motion orthogonal to the surface. This defines dynamics of manifolds in phase space and is defined by the differential equation

$$\frac{\mathrm{d}\boldsymbol{F}_t(\mathbf{y})}{\mathrm{d}t} = \Delta_{\mathbf{y}}$$

Invariant manifolds are then fixed points of this dynamical system. Numerical relaxation methods have been developed to solve this equation known as the film extension of the dynamics [43].

1.1.2 Stochastic model reduction

Recall that model reduction of a dynamical system defined on a phase space $\mathcal{Z}$ seeks a lower dimensional subspace $\mathcal{X} \subset \mathcal{Z}$ that captures the essential or macroscopic dynamics. In the previous section, the inessential dynamics in $\mathcal{Z} \setminus \mathcal{X} := \mathcal{Y}$ were assumed to be negligible which is only valid if the defect of invariance Δ over

$\mathcal{X}$ is sufficiently small. In this section, we describe methods that derive stochastic terms intended to model the effective influence the neglected modes have on the essential modes.

The general setup for such methods can be described by considering the noise-driven differential equation for $z \in \mathcal{Z}$

$$\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} = \boldsymbol{h}(\mathbf{z}) + \boldsymbol{\gamma}(\mathbf{z})\frac{\mathrm{d}\boldsymbol{W}}{\mathrm{d}t}$$

where $\boldsymbol{W}(t)$ is a noise process chosen so that $\mathbf{z}(t)$ is Markovian. Assume that the full dynamics have already been split into essential and inessential variables $\mathbf{x}, \mathbf{y}$, respectively and define the projection operator $P : \mathcal{Z} \to \mathcal{X}$ and the orthogonal complement $\mathcal{Y} = (I - P)\mathcal{Z}$. Introducing coordinates $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$ we obtain the coupled SDEs for dynamics on $\mathcal{X}$ and $\mathcal{Z} \setminus \mathcal{X}$, respectively, written

$$\begin{aligned} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= f(\mathbf{x}, \mathbf{y}) + \alpha(\mathbf{x}, \mathbf{y})\frac{\mathrm{d}U}{\mathrm{d}t} \\ \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= g(\mathbf{x}, \mathbf{y}) + \beta(\mathbf{x}, \mathbf{y})\frac{\mathrm{d}V}{\mathrm{d}t} \end{aligned}$$

where U, V are noise processes. The goal is then to study situations where y variables can be eliminated and an approximate, effective equation for $\mathbf{x}$ can be derived [41].

1.1.2.1 Scale separation and averaging

In the case where the $\mathbf{x}$ and $\mathbf{y}$ dynamics are dominated by linear terms $\mathbf{L}_1$ and $\mathbf{L}_2$ in the form

$$\begin{aligned} \boldsymbol{f}(\mathbf{x}, \mathbf{y}) &= \mathbf{L}_1\mathbf{x} + \boldsymbol{n}_1(\mathbf{x}, \mathbf{y}) \\ \boldsymbol{g}(\mathbf{x}, \mathbf{y}) &= \mathbf{L}_2\mathbf{x} + \boldsymbol{n}_2(\mathbf{x}, \mathbf{y}) \end{aligned}$$

and there exists a large spectral gap between the two linear operators relative to the magnitudes of the nonlinear terms $\boldsymbol{n}_1, \boldsymbol{n}_2$ then it can be proven that an attractive manifold $\mathbf{y} = \boldsymbol{\phi}(\mathbf{x})$ exists such that after an initial relaxation period, the effective dynamics on $\mathcal{X}$ is approximately

$$\frac{\mathrm{d}X}{\mathrm{d}t} = L_1 X + \hat{f}(X, \eta(X))$$

The existence of slow attracting manifolds is considered in [71] and spectral gap arguments are developed in [98, 58, 33, 28].

A similar approach looks for timescale separations between slow and fast variables in SDEs. This is considered in [100] and used to derive a homogenized equation for the fast variables in the case where time-scale separation is not assumed to be infinite as in classic homogenization theory.

In the case where, for fixed $\mathbf{x}$, the $\mathbf{y}$-dynamics don't tend to a fixed point $\mathbf{y} = \boldsymbol{\phi}(\mathbf{x})$, the $\mathbf{y}$-dynamics can be averaged with respect to an ergodic measure to obtain their effective influence on the $\mathbf{x}$-dynamics. Assuming ergodicity of the $\mathbf{y}$-dynamics, Anosov's theorem then states that for fixed x, the slow variables $x(t)$ converge uniformly on any bounded time interval to the solution $X(t)$ of the averaged equation

$$\begin{aligned} \frac{\mathrm{d}X}{\mathrm{d}t} &= F(x) \\ F(\zeta) &= \int_{\mathcal{Y}} f(\zeta, y)\mu_\zeta(\mathrm{d}y) \end{aligned}$$

A detailed account of this averaging method is found in [85] and applied to a model set of equations in [75]. Extensions of this approach to more general assumptions are given in [62], [12].

1.1.2.2 Projection onto fast variables

A rigorous technique is for projection of a dynamical system onto fast-variables is given by the Mori-Zwanzig formalism [67], [104]. A discussion of the Mori-Zwangig technique in [35] takes a Hamiltonian perspective. Coordinates in phase space are given by $\Gamma = \{p_n, q_n\}$. A Hamiltonian $H(\Gamma)$ defines a flow in phase space via

$$\frac{\mathrm{d}\Gamma}{\mathrm{d}t} = [\Gamma, H] = iL\Gamma$$

where $[X, Y] = \sum \left(\frac{\partial X}{\partial q} \frac{\partial Y}{\partial p} - \frac{\partial X}{\partial p} \frac{\partial Y}{\partial q} \right)$. The Zwanzig projection operator operates on function in the $6N$-dimensional phase space of N point particles and projects onto the linear subspace of "slow" phase space functions. A special susbset of these functions is an enumerable set of "slow variables" $A(\Gamma) = \{A_n(\Gamma)\}$.

Application of the operator identity

$$e^{iLt} = e^{i(1-P)Lt} + \int_0^t \mathrm{d}s e^{iL(t-s)} PLe^{i(1-P)Ls}$$

to $(1-P)iLA$, allows for the derivation of a projected stochastic differential equation for the slow variables

$$\frac{\mathrm{d}}{\mathrm{d}t}A(t) = \Omega A(t) + \underbrace{\int_0^t \mathrm{d}s K(s)A(t-s)}_{\text{memory term}} \quad + \quad \underbrace{F(t)}_{\text{random force}}$$

where $K(t)$ represents the memory function and $F(t)$ the random force. Hence the change in the slow variables depends on their current state, their values at previous times, and a random force that depends on the part of the dynamics orthogonal to $A(t)$.

1.1.2.3 Extraction of coarse-grained Markov process

This approach is concerned with the extraction of small and finite state Markov chains from SDEs or larger Markov chains. In the case of SDEs [54], a natural class of model problems exhibiting this transition behavior are gradient systems with additive noise:

$$\frac{\mathrm{d}X}{\mathrm{d}t} = -\nabla V(X) + \sigma \frac{\mathrm{d}W}{\mathrm{d}t}$$

where V is a double-well potential. If σ is small then large deviation theory may be used to derive a two-state Markov chain describing transitions between the wells of V. This can also be carried out in the case of large Markov chains where the linear operator defining the forward equation for the chain is used to partion the state space into coarser subspaces [36].

1.1.3 Dimensionality reduction of data

Dimensionality reduction can be viewed as a special case of model reduction where high-dimensional point clouds are the only type of data mapped to a lower-dimensional space. Vector fields and other components of dynamical systems are not considered. Nonlinear manifold approaches are more applicable to high-dimensional problems in this point cloud setting.

A standard approach is to seek a lower dimensional linear subspace on which the points approximately reside. Principal component analysis (PCA)[48] is the most common method which finds a subspace that minimizes the error between the original data and its projection onto the subspace. This is essentially the same method used in POD and will not be recounted here in detail. PCA can be extended to a nonlinear method by using a nonlinear mapping of the data $\Phi : \mathbb{R}^{D_i} \to \mathbb{R}^{D_f}$ into a high-dimensional feature space

$\mathbb{R}^{D_f}$ such that

$$k(\mathbf{x}, \mathbf{y}) = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y}))$$

for some kernel $k : \mathbb{R}^{D_i} \times \mathbb{R}^{D_i} \to \mathbb{R}$. Carrying out standard PCA in the high-dimensional space yields nonlinear encoding in the original space of input data [88].

As discussed in previous sections, linear subsaces are limited in the sense that data may reside on low-dimensional, nonlinear manifolds which linear subspaces will fail to capture. A natural bridge from linear to nonlinear spaces is provided by the Locally Linear Embedding (LLE) procedure [84]. Manifolds can be approximated locally by linear patches such that the entire manifold can be thought of as locally linear regions stitched together with the linear structure varying across patches. The LLE procedure uses this idea by characterizing the local geometry of patches with linear coefficients which reconstruct each data point from linear combinations of its neighbors.

While LLE seeks to preserve the local linearity of a manifold, several approaches exist which use graph based representations of data to extract different local geometric properties of the data and preserve them in the low dimensional embedding. Isomap [95] seeks to preserve distance structure extracted from neighborhoods of a point. It extends the idea of multidimensional scaling where geodesic distances extracted from a weighted graph are used to determine the low-dimensional embedding. Another example is given by Laplacian Eigenmaps [19] which uses the connection between the graph Laplacian, the Laplace Beltrami operator on a manifold, and the heat equation to provide a nonlinear embedding which preserves local geometric structure.

1.2: DEEP NEURAL NETWORK MODELS

Finding a low-dimensional nonlinear manifold for high-dimensional data representation can be viewed as constructing an appropriate nonlinear transformation into a low-dimensional space. While a nice theory exists for linear maps between finite dimensional spaces, no such theory exists for nonlinear transformations. Instead, we can look for a useful heuristic for developing nonlinear model reduction procedures. One such heuristic comes from the observation that single hidden layer neural networks carry out principal component analysis (PCA) when trained to find the optimal solution to maximization of linearly transformed variances $\mathbb{E}[(\boldsymbol{w}(i)^T \boldsymbol{x})^2]$ [53]. Here $\boldsymbol{w}(1), \ldots, \boldsymbol{w}(M)$ are the weight vectors of the single weight matrix and $\boldsymbol{x}$ is a mean-zero data vector. By altering aspects of the neural network such as the the loss $\mathbb{E}[(\boldsymbol{w}(i)^T \boldsymbol{x})^2]$ and

network architecture, we can obtain nonlinear generalizations of PCA which can be viewed as extracting a type of nonlinear mode from the data.

1.2.1 Overview of neural network models

The foregoing discussion suggests that deep learning [60] provides suitable models for nonlinear dimesion-reducing transformations and this will ultimately be the model reduction approach taken in this thesis. Hence, it will be useful to provide a brief description of neural network models.

Neural network architectures can typically be visualized as graphs with nodes connected by edges that represent functional dependence between them. Recurrent neural networks can contain cyclic connections that allow previous outputs to be used as inputs. On the other hand, feedforward networks can be represented by a directed, acyclic graph where information flows only in one direction.

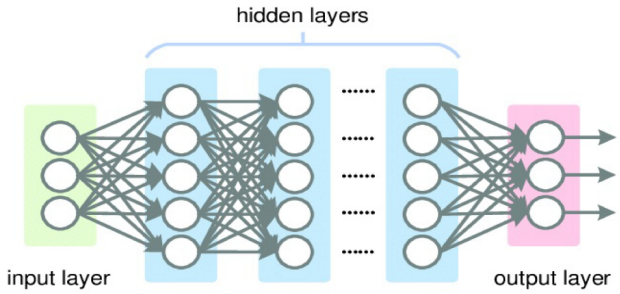


Figure 1.2: Feedforward neural network structure.

Feedforward networks often have a straightforward mathematical representation as a sequence of composed functions of the form

$$\begin{aligned} \boldsymbol{F}(\mathbf{x}, \mathcal{W}) : \mathbb{R}^n \times \mathbb{R}^{D_w} &\rightarrow \mathbb{R}^m \\ \mathbf{x} &\mapsto \boldsymbol{f}^L \circ \boldsymbol{f}^{L-1} \circ \cdots \circ \boldsymbol{f}^1 \circ (\boldsymbol{x}) \end{aligned}$$

where the $\boldsymbol{f}^i$ represent the operations carried out by each of the L layers of the network $\boldsymbol{F}$ parametrized by weights $\mathcal{W} \in \mathbb{R}^{D_w}$.

Consideration of different layer operations and overall network architecures leads to a variety of classes of neural network models. These classes are also distinguished by the specific types of problems whose structure lends itself to properties of the particular network class. Several important examples are presented in the following sections.

1.2.1.1 Convolutional neural networks

Convolutional neural networks (CNNs) [82] are a type of feedforward neural network loosely inspired by visual cortex structure and designed for image classification [59]. In particular, they use convolutional layers that can pick up spatial correlations among local groups of image pixels. Pooling layers are also used to subsample inputs and obtain lower resolution images.

Convolutional layers play the role of feature extractors and learn feature representations from input images. This is done sequentially so that complex feature extractors can be learned. The nodes or neurons in the convolutional layers form a feature map and are related more precisey by

$$Y_k = f(W_k * x)$$

where x represents the input image, W_k denotes the convolutional filter for the kth feature map, and f is some nonlinear activation function applied component-wise. Pooling layers reduce the spatial resolution of the feature maps. For example, max pooling selects the largest element within each receptive field.

CNNs effectively leverage the local stationarity of natural images at mutiple scales using convolution operations and provide translation invariance through pooling layers [25, 24, 80].

1.2.1.2 Neural networks for regression

A typical regression DNN structure is simply a nonlinear function which is the repeated composition of a number of simpler functions referred to as the layers of the network. Nonlinearity is achieved by having alternating layers of linear or affine transformations and simple element-wise nonlinear activation functions. Hence a DNN $\boldsymbol{F}(\boldsymbol{x}, \mathcal{W}) : \mathbb{R}^n \times \mathbb{R}^{D_w} \rightarrow \mathbb{R}^m$ is given by

$$\boldsymbol{f}^L \circ \boldsymbol{W}^L \circ \boldsymbol{f}^{L-1} \circ \boldsymbol{W}^{L-1} \circ \cdots \circ \boldsymbol{f}^1 \circ \boldsymbol{W}^1(\boldsymbol{x}) \tag{1.7}$$

where $\boldsymbol{W}^i$ is the linear transformation from layer i defined by a matrix of weights, and $\boldsymbol{f}^i$ is a nonlinear activation function applied elementwise

$$\boldsymbol{f}^i(\boldsymbol{x}) = \begin{bmatrix} f(\boldsymbol{x}_1) \\ \vdots \\ f(\boldsymbol{x}_d) \end{bmatrix}$$

Common choices for the activation function are the piece-wise linear function ReLu and logistic functions.

The network is trained to learn an input-output mapping from training data $\mathcal{T} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N_T}$ in the form of input-target pairs where learning amounts to minimizing a cost function which measures the error of the network's performance on the training examples. The training process is carried out through gradient descent $\mathcal{W}^{n+1} = \mathcal{W}^n - \eta \nabla Q_i(\mathcal{W}^n)$ using a cost-function of the form $Q(\mathcal{W}) = \frac{1}{n} \sum_{i=1}^{N_T} Q_i(\mathcal{W})$, with i ranging over the training examples. To improve efficiency, stochastic gradient descent [90] can be used which computes a random subset of the sum of gradients as an approximation to the true gradient over all training examples.

Autoencoders are a class of feedforward neural networks whose architecture makes them suitable for dimension reduction by encoding data into a low-dimensional space through a nonlinear transformation [47]. Autoencoders consist of an initial set of layers defining an encoder which maps high-dimensional inputs into a low-dimensional space. The final set of layers form a decoder which maps the low-dimensional data back to the orginal high-dimensional space. The autoencoder $\boldsymbol{A}(\boldsymbol{x}, \mathcal{W}) : \mathbb{R}^D \rightarrow \mathbb{R}^D$ is trained to minimize the

reconstruction error

$$\min_{\mathcal{W}} \frac{1}{N_T} \sum_{i=1}^{N_T} \|\boldsymbol{x}_i - \boldsymbol{A}(\boldsymbol{x}_i, \mathcal{W})\|^2 = \min_{\mathcal{W}} \frac{1}{N_T} \sum_{i=1}^{N_T} \|\boldsymbol{x}_i - \boldsymbol{d}_{\mathcal{W}} \circ \boldsymbol{e}_{\mathcal{W}}(\boldsymbol{x}_i)\|^2$$

across data points $\mathcal{T} = \{\boldsymbol{x}_i\}_{i=1}^{N_T}$ forming the training examples. Here, $\boldsymbol{e}_{\mathcal{W}} : \mathbb{R}^D \to \mathbb{R}^d$ and $\boldsymbol{d}_{\mathcal{W}} : \mathbb{R}^d \to \mathbb{R}^D, d \ll D$ are the encoder and decoder maps, respectively, which depend on the collection of weights $\mathcal{W}$ and comprise the autoeconder $\boldsymbol{A}$.

The autoencoder framework for dimensionality-reduction provides a nonlinear analogue of the linear model reduction procedure in section 1.2. Linear model reduction encodes high-dimensional data through a linear map $\boldsymbol{B}^T$, where $\boldsymbol{B} = [\boldsymbol{b}_1 \quad \cdots \quad \boldsymbol{b}_d]$ is an orthonormal basis for a d-dimensional subspace of $\mathbb{R}^D$. This map represents the coordinates of projection onto the linear subspace spanned by the columns of $\boldsymbol{B}$ which forms the corresponding linear decoder mapping back to the orginal space. The autoencoder carries out a nonlinear version of this procedure (figure 1.3).

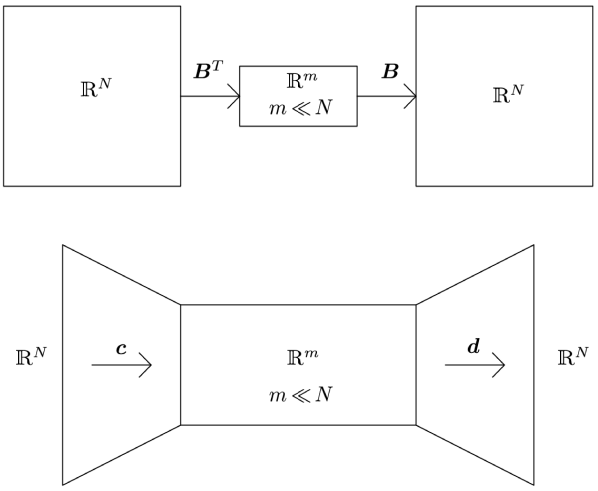


Figure 1.3: Comparison of linear model reduction (top) with nonlinear model reduction using an autoencoder (bottom).

1.2.2 Machine learning approaches to model reduction

Model reduction approaches that use machine learning typically train neural networks to act as low-dimesional surrogates for high-dimensional nonlinear functions. Surrogates are trained to emulate nonlinear functions using data generated from physics-based simulations in an offline phase. The learned functions are used in the online phase to carry out simulations at a reduced computational cost.

This includes learning reduced linear operators [79] or the map between input paramters for a physical system model and reduced (POD) expansion coefficients [94], [31]. In the case of reduced models, neural networks can also be used to learn a closure term that models the effect of neglected modes on the dominant, reduced modes [101]. Neural networks can also act as surrogates outside of POD approaches such as the residual minimization procedure component of implict integration schemes [52].

1.3: MOLECULAR DYNAMICS

This thesis considers MD simulations as a stochastic physics model to serve as a testbed for the acceleration procedure. Molecular dynamics is a classical approach to modeling molecular physics where particles are represented as points rather than wave functions. This can be viewed as a homogenization over the quantum scale of description to obtain a classical physics model that avoids the computational complexity of quantum mechanical approaches like density functional theory. Molecular dynamics allows for the simulation of larger molecules such as biological proteins, which are composed of numerous amino acid residues and contain on the order of 10^3 atoms. To mimic the effects of a thermal bath, stochastic terms are present and typically have Gaussian white noise statistics. Thus, the overall MD system is of the form

$$\mathrm{d}\boldsymbol{X} = \boldsymbol{F}(\boldsymbol{X})\mathrm{d}t + \mathrm{d}\boldsymbol{B}$$

where $\boldsymbol{X}(t) \in \mathbb{R}^n$ denotes the atom positions with n large. The interatomic forces and thermal background forces are given by $\boldsymbol{F}(\boldsymbol{X})$ nonlinear and $\mathrm{d}\boldsymbol{B}$, respectively. The time step δt of direct numerical simulation is small $\delta t \sim \mathcal{O}(10^{-12})$s and protein folding time is much larger $T_f \sim \mathcal{O}(10^{-3})$s [32], [69] resulting in $\mathcal{O}(10^9)$ time steps necessary to observe folding events. This becomes infeasible to carry out, even for smaller proteins, and necessitates the development of coarse-graining procedures to acheive a reduction in the dimension of the problem and thereby decrease the simulation time necessary to folding.

1.3.1 MD potentials

Atoms in MD models interact through nonlinear, conservative forces defined by an empirically validated potential designed to generate realistic electrostatic interactions between pairs of atoms and enforce geometric constraints among larger groups of atoms. A widely used force field is given by the AMBER potential [29, 97, 76] which is the form used for MD simulations in this thesis. It consists of pairwise terms and many body terms for groups of three and four atoms. It has the form

$$\begin{aligned} U(\boldsymbol{r}) &= \sum_{\text{bonds}} k_{ij}^{b}(r_{ij}-r_0)^2 \\ &+ \sum_{\text{angles}} k_{ijk}^{a}(\theta_{ijk}-\theta_0)^2 \\ &+ \sum_{\text{torsions}} k_{ijkl}[1+\cos(n_{ijkl}\phi_{ijkl}-\phi_0)] \\ &+ \sum_{j=1}^{N-1}\sum_{i=j+1}^{N} f_{ij}\left\{\epsilon_{ij}\left[\left(\frac{r_{ij}^0}{r_{ij}}\right)^{12}-2\left(\frac{r_{ij}^0}{r_{ij}}\right)^{6}\right]+\frac{q_iq_j}{4\pi\epsilon_0 r_{ij}}\right\} \end{aligned} \tag{1.8}$$

where $r_{ij}, \theta_{ijk}, \phi_{ijkl}$ give the distance between atoms i, j, angle between atoms i, j, k and dihedral angle of atoms i, j, k, l, respectively and r_0, θ_0, ϕ_0 their equilibrium values [76, 29]. A geometric depiction of what the various terms represent is provided in figure 1.4.

MD simulations typically sample from the canonical ensemble which represents the probability distribution over possible states of a mechanical system in thermal equilibrium with a heat bath at a fixed temperature. The amount of substance, volume and temperature are conserved. Thermostat algorithms are employed to add or remove energy from the MD simulation to approximate the canonical ensemble and thereby introduce stochasticity into the simulation.

1.3.2 Numerical integration

The differential equations defining MD are numerically integrated with sympletic schemes that involve a thermostat element for maintaining a set temperature and modeling unresolved physics through a dissipative force. A common numerical method is the middle scheme which is simply the leapfrog method with an extra thermostat step. In the middle scheme, integration in one time step Δt can be split into three parts for updating the coordinates, momenta and thermostat.

(a) Angle and bonds (b) Torsions

Figure 1.4: Geometry of MD potentials

The equations of motion are expressed as

$$\begin{bmatrix} \mathrm{d}\boldsymbol{x}_t \\ \mathrm{d}\boldsymbol{p}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{M}^{-1}\boldsymbol{p}_t \\ 0 \end{bmatrix} \mathrm{d}t + \begin{bmatrix} 0 \\ -\nabla_{\boldsymbol{x}_t} U(\boldsymbol{x}_t) \end{bmatrix} \mathrm{d}t + [\text{thermostat}] \tag{1.9}$$

where U is the interatomic potential and $\boldsymbol{x}, \boldsymbol{p}$ are the coordinates and momenta of atoms, respectively. This expression of the dynamics is not amenable to mathematical analysis. Instead, it is more useful to adopt the point of view of the forward Kolmogorov equation expressing the evolution of the density distribution in phase space $\rho(\boldsymbol{x}, \boldsymbol{p})$

$$\begin{aligned} \frac{\partial}{\partial t}\rho &= \mathcal{L}\rho \\ &= (\mathcal{L}_x + \mathcal{L}_p + \mathcal{L}_T)\rho \end{aligned}$$

$$\frac{\partial}{\partial t}\rho = \mathcal{L}\rho$$

where we have

$$\begin{aligned}\mathcal{L}_x\rho &= -\boldsymbol{p}^T\boldsymbol{M}^{-1}\nabla_{\boldsymbol{x}}\rho \\ \mathcal{L}_p &= \nabla_{\boldsymbol{x}}U(\boldsymbol{x})\cdot\nabla_{\boldsymbol{p}}\rho\end{aligned}$$

and the definition of $\mathcal{L}_T$ depends upon the particular thermostat used. Thus we have the three phase space propagators for time interval Δt as $e^{\mathcal{L}_{\boldsymbol{x}}\Delta t}$, $e^{\mathcal{L}_{\boldsymbol{p}}\Delta t}$, and $e^{\mathcal{L}_{\boldsymbol{T}}\Delta t}$. The propagation for each time step for velocity Verlet is given by

$$e^{\mathcal{L}\Delta t} \approx e^{\mathcal{L}^{\mathrm{VV}}_{\mathrm{middle}}\Delta t} =$$

The above decomposition of the Louiville operator gives a symplectic integration scheme. This is desired as the time evolution of Hamilton's equation conserves the symplectic two-form $dp \wedge dq$ defined on phase space. A symplectic integrator conserves a slightly perturbed Hamiltonian as it consists of the composition of several symplectomorphims, transformations which preserve area in phase space.

A common procedure to simulate thermal equilibrium with a heat bath is the Andersen thermostat. In this system, each particle stochastically collides with a fictitious heat bath and the momentum of the particle is sampled from the Maxwell-Boltzmann momentum distribution such that

$$\boldsymbol{p}^{(j)} \leftarrow \sqrt{\frac{m_j}{\beta}}\boldsymbol{\theta}_j, (j = 1, \ldots, N)$$

$$\text{if } \mu_j < 1 - e^{-\nu\Delta t}$$

where ν is the collision frequency, $\boldsymbol{\theta}_j$ i.i.d standard normal distribution, m_j mass for the jth atom, μ_j, a uniformly distributed random number in $(0, 1)$.

1.3.3 Alpha helix formation as an elementary example of protein folding

The alpha helix is a common form of secondary structure among proteins where hydrogen bonds between amino acid residues at differing locations along the sequence cause the formation of a stable helical shape. The relative simplicity of the alpha helix makes it a suitable benchmark for comparison of standard MD and the DNN accelerated MD approach investigated here.

The geometry of the alpha helix is comprised of amino acids arranged in a right-handed helical structure with 3.6 residues per helical turn and a pitch of 0.54 nm. The hydrogen bond inducing the helix is between the N-H group of an amino acid and C=O group of the amino acid four residues earlier. The backbone dihedral angles (ϕ, ψ) of an alpha helix are typically around $(-60°, -45°)$.

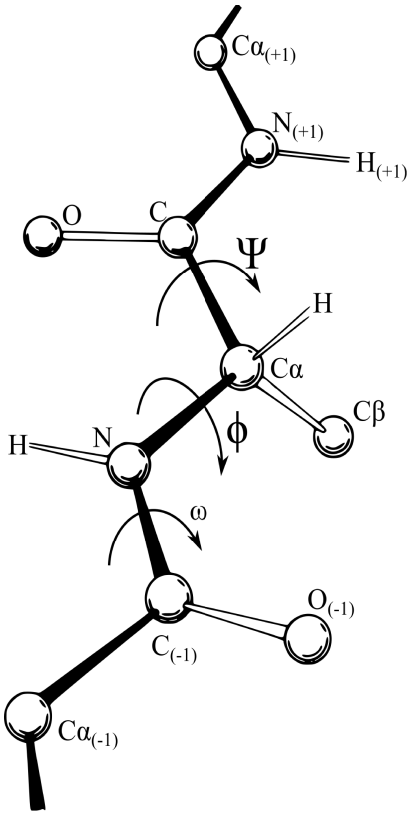


Figure 1.5: Dihedral angles of a protein backbone.

There are a variety of models for alpha helix formation based on statistical mechanics. Helix-coil theory models helix formation as a two step process: helix nucleation can occur at random locations, and helix propogation, the addition of additional helical segments, can take place only after a helical nucleus has formed [89]. This approach represents the transition between a disordered protein and alpha-helix structure where each residue can be in one of two states: a helical state (h) or random coil state (c) such that there are 2^l configurations in the set $\mathcal{X}$ of states of a peptide of length l. A configuration is represented as a binary vector $X = (x_1, \ldots, x_l)$. Given a peptide sequence $R = (R_1, \ldots, R_l)$ each configuration is assigned an energy

$$U(X, R) = \sum_{i=1}^{l} x_i \Delta G_{R_i}$$

where ΔG_{R_i} is the change in free energy for position i going from coil to helical. This induces a Boltzmann-Gibbs distribution on $\mathcal{X}$

$$P(X \in \mathcal{X}|R) = Z^{-1}e^{-\beta U(X,R)}$$

where Z is the partition function

$$\sum_{X \in \mathcal{X}} \exp(-\beta U(X, R))$$

The Lifson-Roig version of the helix-coil models uses weight parameters u, v, w defined by u: coil states, v: helix states with one or more coil neighbors, and w: helix states with two helical neighbors. The intution being that v is a helix nucleation parameter and w a helix extension parameter. The probability of the example configuration

$$\begin{array}{cccccccc} c & c & h & h & h & h & c & c \\ u & u & v & w & w & v & u & u \end{array}$$

can be computed as the product $Z^{-1}u^4v^2w^2$. This simply reflects the number of occurences of the various terms.

Then the helicity of a configuration is $h(X) = \frac{1}{l}\sum_{i=1}^{l} x_i$ and the helicity of a peptide is the expectation of $h(X)$ over the configuration space $\mathcal{X}$:

$$\mathcal{H}(R) = Z(R, T, \theta)^{-1} \sum_{X \in \mathcal{X}} h(X)e^{-\beta U(X,R)}$$

Predicted values of quanties such as the mean number of helical segments of two or more residues can be computed similarly and compared with experimental data [87].

1.3.4 Alpha helix nucleation

Helix nucleation times vary in the literature. In [93], simulations of various AMBER potentials predict initial nucleation to occur on the tens of picoseconds time scale while in [34], it is reported to be on the tens of nanosecond scale. An experimental investigation [32] revealed alpha helix nucleation to occur on the millisecond time scale. Despite the disparity in reported nucleation times, nucleation preceeds formation of a full alpha helix structure and serves as a suitable landmark with which to test DNN acceleration techniques.

1.3.5 Coarse-graining approaches for MD

A variety of linear projection methods have been applied to construct reduced, also known as coarse-grained, models of MD. For example, the MARTINI force field for MD is based on a 4-to-1 averaging operator [65] and there exist a number of other related methods which derive reduced systems using averaging transfromations [78], [2]. This includes the popular Go coarse-grained model [72] which can fail to produce the correct behavior of the molecular system [57]. Some approaches consider more general linear mappings from fine scale to coarse scale variables [50], [70], [83], [49].

1.4: CONCLUSION AND ORIGINAL CONTRIBUTIONS

The previous sections discussed the limitations of linear model reduction when applied to nonlinear systems. These linear model reduction procedures are typically carried out on deterministic systems leaving the acceleration of stochastic systems an important open problem. This motivates the consideration of nonlinear model reduction procedures for both deterministic and stochastic systems which is the central focus of the contributions in this thesis. Molecular dynamics will be used as a test-bed for the acceleration techniques developed as it provides a challenging nonlinear, stochastic system with mesoscale behavior defined by protein folding. This thesis makes several original contributions to deterministic and stochastic model reduction procedures which are listed below:

- A successful nonlinear model reduction procedure in section 2.2 based on nonlinear dimension reduction and DNN regression of a mesoscale forward map.
- Development of a workflow described in 2.4.2 for automating the process of carrying out computationally expensive MD simulations with a focus on mesoscale behavior.
- Initialized the study of a stochastic model reduction procedure in 3.2 based on the extrapolation of moments and provided a proof of concept by using it to achieve acceleration of stochastic gradient descent.
- Began development of a theoretical framework in chapter 4 for carrying out stochastic model reduction on manifolds of probability distributions by geodesic projection onto lower-dimensional subspaces.

In the subsequent chapters, these contributions are presented in detail with relevant theoretical motivation, descriptions of the various computational procedures implemented, and finally the results of applying these model reduction methods to both deterministic and stochastic dynamical systems.

CHAPTER 2: NONLINEAR MODEL REDUCTION OF DETERMINISTIC DYNAMICS THROUGH DEEP NEURAL NETWORKS

2.1: INTRODUCTION

The previous chapter concluded that DNNs provide a class of nonlinear transformations suitable for nonlinear model reduction. Autoeconder DNNs can achieve dimension reduction of high-dimensional data by learning an encoding operation into low-dimensional space and its corresponding decoding operation. Feedforward neural networks provide functions for nonlinear regression and can be used for model reduction by approximating complex, nonlinear functions in an offline phase. The resulting networks can then act as surrogates for functions that can be evaluated at a reduced online cost.

Before applying DNNs to the problem of model reduction for stochastic systems, it is necessary to establish that DNNs can carry out model reduction in the deterministic case where stochastic behavior is not accounted for by the procedure. To that end, consider a differential equation of the form

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = \boldsymbol{F}(\mathbf{x}), \mathbf{x}(0) = \mathbf{x}_0 \tag{2.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^D$, with D large, and $\boldsymbol{F}(\mathbf{x})$ nonlinear. We would like to use DNNs to capture the slow, dominant modes of the system which reflect the dynamics over large timescales. To gain insight as to how DNNs can extract dominant modes of (2.1), consider the simple case of a linear initial value problem

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{A}\mathbf{x}, \mathbf{x}(0) = \mathbf{x}_0 \tag{2.2}$$

where $\mathbf{A}$ is symmetric, positive definite and admits a unitary diagonalization $\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$. Suppose that the eigenvalues $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_D \geqslant 0$ comprising the diagonal of $\boldsymbol{\Lambda}$ exhibit a spectral gap

$$\lambda_1 \geqslant \cdots \geqslant \lambda_d \gg \lambda_{d+1} \geqslant \cdots \geqslant \lambda_D$$

where the first d eigenvalues are large compared to the remaining ones. The eigenmodes of the system are given by the eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_D$ with the first d eigenvectors comprising the dominant modes of the system. The solution of the system is given by the matrix exponential

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}_0 = \mathbf{U}e^{\mathbf{\Lambda}t}\mathbf{U}^T\mathbf{x}_0$$

Hence, the forward propagation operator $\mathcal{F}_{t_1,t_2} : \mathbb{R}^n \to \mathbb{R}^n$ mapping $\mathbf{x}(t_1) \in \mathbb{R}^D$ to the solution of (2.2) at time t_2 is given by

$$\begin{aligned}
\mathcal{F}_{t_1,t_2}(\mathbf{x}_0) &= [e^{\lambda_1\tau}\mathbf{u}_1\mathbf{u}_1^T + \cdots + e^{\lambda_n\tau}\mathbf{u}_D\mathbf{u}_D^T]\mathbf{x}_0 && (2.3)\\
&= [e^{\lambda_1\tau}\mathbf{u}_1\mathbf{u}_1^T + \cdots + e^{\lambda_d\tau}\mathbf{u}_d\mathbf{u}_d^T]\mathbf{x}_0 && (2.4)\\
&+ [e^{\lambda_{d+1}\tau}\mathbf{u}_{d+1}\mathbf{u}_{d+1}^T + \cdots + e^{\lambda_n\tau}\mathbf{u}_D\mathbf{u}_D^T]\mathbf{x}_0 && (2.5)
\end{aligned}$$

where $\tau := t_2 - t_1$. Note that the first term on the right hand side of (2.5) is dominant as $t \to \infty$. This suggests a model reduction procedure based on the dominant modes $\mathbf{u}_1, \ldots, \mathbf{u}_d$. First, the initial data $\mathbf{x}_0$ is projected onto the dominant modes to provide reduced coordinates $u_1, \ldots, u_d$ defined by $u_1 = \mathbf{u}_1^T\mathbf{x}_0, \ldots, u_d = \mathbf{u}_1^T\mathbf{x}_0$. This can be viewed as the linear form of an encoding transformation onto dominant modes of the system. By neglecting the non-dominant modes $\mathbf{u}_{d+1}, \ldots, \mathbf{u}_n$, the forward map (2.5) can be written as a function

$$\tilde{\mathcal{F}}_{t_1,t_2}(\tilde{\mathbf{u}}) = \begin{bmatrix} u_1 e^{\lambda_1\tau} & \cdots & u_d e^{\lambda_d\tau} \end{bmatrix}^T$$

advancing the dominant mode coordinates $\tilde{\mathbf{u}} = (u_1, \ldots, u_d)$ forward in time from t_1 to t_2. The dominant modes at t_2 can be lifted back to the original space $\mathbb{R}^n$ by

$$\mathbf{x}(t_2) \approx \mathbf{U}_d\tilde{\mathcal{F}}_{t_1,t_2}(\tilde{\mathbf{u}})$$

where $\mathbf{U}_d = [\mathbf{u}_1 \cdots \mathbf{u}_d]$. This can similarly be viewed as the linear form of a decoding transformation from the dominant modes to the full system.

The above discussion is restricted to a linear context but outlines a general model reduction approach that can be extended to a nonlinear procedure by replacment of the various linear operations with nonlinear analogues. The procedure consists of first encoding the system into a low-dimesional space of coordinates.

This can be done linearly by PCA or nonlinearly by an autoencoder or nonlinear dimension reduction technique such as LLE or Isomap. The reduced coordinates are then propogated forward in time and transformed back to the orginal space by an appropriate decoding operation.

Recall that linear model reduction procedures typically carry out POD to determine a dominant linear subspace $\mathcal{V} \subset \mathbb{R}^D$, $\dim(\mathcal{V}) = d \ll D$ defined by the column space of $\mathbf{V}$. Projection onto this subspace comprises the encoding operation onto reduced coordinates. To propogate these reduced coordinates forward in time, the vector field $\boldsymbol{F}(\mathbf{x})$ is projected onto $\mathcal{V}$ yielding an ODE system on $\mathbb{R}^d$. In the linear case given by (2.2), the projected vector field $\tilde{\mathbf{A}}(\mathbf{y}) = \mathbf{V}^T\mathbf{A}\mathbf{V}(\mathbf{y})$ is simply a low-dimensional linear operator $\tilde{\mathbf{A}} \in \mathbb{R}^{d\times d}$. In the nonlinear case the projected vector field is of the form $\widetilde{\boldsymbol{F}}(\mathbf{y}) = \mathbf{V}^T\boldsymbol{F}(\mathbf{V}\mathbf{y})$ which still requires $\mathcal{O}(D)$ operations to compute and prevents any significant acceleration. A number of approaches to this problem seek to approximate $\widetilde{\boldsymbol{F}}$ by a lower-dimensional function that can be evaluated efficiently. DNNs provide the capability of function regression and are a natural consideration for approximating the nonlinear, reduced vector field. If we consider an encoding procedure in the form of a nonlinear map onto general submanifold coordinates, it is no longer straightfoward to find the projection of a linear or nonlinear vector onto these submanifold coordinates. Hence, we instead take the approach of carrying out regression on the forward map for the reduced coordinates to avoid the complex geometry of the underlying submanifold.

2.2: NONLINEAR MODEL REDUCTION PROCEDURE

This section presents the details of the deterministic DNN model reduction procedure. An appropriate encoding transformation is introduced that acheives sufficient compression of snapshot data sampled from DNS of the dynamical system. Encoding these snapshots provides samples of the evolution of the reduced coordinates of the system. A mesoscale is introduced and DNNs are used to perform regression of the forward map of the reduced coordinates aross some mesostep. The system can be pushed forward in time at reduced computational cost by iteratively applying the learned foward map and then decoding the resulting reduced coordinates to obtain the state of the full system at a future time.

Consider an ODE system of the form (2.1) with exact solution $\mathbf{x}(t) \in \mathbb{R}^D$ approximated by direct numerical simulation where the time step Δt is chosen to ensure stability. The interval Δt defines the microscale of the problem and is assumed to be small with respect to a chosen mesoscale $\Delta\tau = N_{\text{meso}}\Delta t$ on which the large scale behavior of interest is observable. The resulting sequence of states, with respect to initial

condition $\mathbf{x}_0$, is given by $\mathbf{x}(t_i), i = 1, \ldots, N_t$ where $t_i = i(\Delta t)$. The first stage of the procedure consists of subsampling the states of the system in the form of a collection of n_s snapshots $\mathbf{x}(t_{i_1}), \ldots, \mathbf{x}\left(t_{i_{n_s}}\right)$. A dimension reduction method is chosen based on the data to define an encoding function $\boldsymbol{E} : \mathbb{R}^D \to \mathbb{R}^d$, $d \ll D$ which is applied to the sequence of high-dimensional states $\mathbf{x}(t_i)$ to generate samples of the reduced dynamics $\mathbf{y}(t_i) = \boldsymbol{E}(\mathbf{x}(t_i)), i = 1, \ldots, N_t$. The encoding function is paired with a corresponding decoding function $\boldsymbol{D} : \mathbb{R}^d \to \mathbb{R}^D$ used to approximately reconstruct high-dimensional data from reduced coordinates. While this provides dimension reduction of phase space point data, it remains to implement model reduction of the dynamical component of the system. To achieve this, DNN regression is carried out to approximate the mesoscale forward map which advances reduced positions into the future by one mesostep $\Delta\tau$. This suggests the training data should be sampled from the set of pairs $\{(\mathbf{y}(t_i), \mathbf{y}(t_{i+N_{\text{meso}}})\}_{i=1}^{N_t}$. The training process to determine weights $\boldsymbol{\mathcal{W}}$ for the forward map approximant $\boldsymbol{F}_{\Delta\tau}(\mathbf{y}, \boldsymbol{\mathcal{W}}) : \mathbb{R}^d \to \mathbb{R}^d$ is carried out through stochastic gradient descent $\boldsymbol{\mathcal{W}}^{n+1} = \boldsymbol{\mathcal{W}}_M^n - \eta \nabla Q(\boldsymbol{\mathcal{W}}^n)$ using a mean-squared error cost-function of the form $Q(\boldsymbol{\mathcal{W}}) = \frac{1}{N_t} \sum_{i=1}^{N_t} Q_i(\boldsymbol{\mathcal{W}})$ where Q_i defines the contribution of the ith data point to the cost. Iterative application of the learned mesoscale forward map $\boldsymbol{F}_{\Delta\tau}$ allows for time-evolution of the system starting from an initial state $\mathbf{y}(\tau_M)$ in the form

$$\mathbf{y}(\tau_{M+1}) = \boldsymbol{F}_{\Delta\tau}(\mathbf{y}(\tau_M)), \ldots, \mathbf{y}(\tau_{M+k}) = \boldsymbol{F}_{\Delta\tau}^k(\mathbf{y}(\tau_M))$$

Application of the decoding function yields approximate corresponding states of the full system

$$\mathbf{x}(\tau_{M+1}) \approx \boldsymbol{D} \circ \boldsymbol{F}_{\Delta\tau}(\mathbf{y}(\tau_M)), \ldots, \mathbf{x}(\tau_{M+k}) \approx \boldsymbol{D} \circ \boldsymbol{F}_{\Delta\tau}^k(\mathbf{y}(\tau_M))$$

at reduced computational cost compared to DNS.

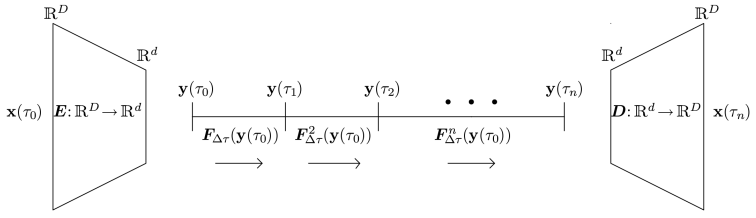


Figure 2.1: Deterministic model reduction based on DNNs.

2.3: APPLICATION TO FPU SYSTEM

Before applying the acceleration procedure to a high-dimensional, stochastic system such as MD, it is important to establish that DNNs are able to extract a reduced description from a simple model that is consistent with known features. We would like a system that exhibits large, mesoscale features understood analytically and that can be compared to a numerically obtained reduced description. MD potentials typically include a harmonic term for each bonded pair among other high-order nonlinear terms that constrain the motion of atoms in various ways. MD potentials can then be viewed as inducing harmonic motion on a network of bonded atoms superposed with additional forces defined by the higher-order nonlinear terms. This suggests a relevant test model is the 1-dimensional chain of nonlinear oscillators originally simulated in the Fermi-Pasta-Ulam (FPU) computer experiments [102, 23]. At first glance, these two dynamical systems have quite different mathematical properties with the FPU chain being deterministic, integrable, and Hamiltonian while MD represents a stochastic, Langevin system. But these properties are not relevant as the goal is to show that DNNs can extract the large scale, dominant modes of the dynamical system from a reduced description of snapshot data. Both the FPU and MD systems exhibit correlated motion of many individual components occurring over a large time scale and hence provide a relevant comparison. The large scale features in the FPU system consist of nonlinear traveling waves which provide an analytically understood large-scale feature that can be used to validate the model reduction procedure.

2.3.1 Definition

The model is defined by N oscillators comprising a string of length l whose equilibrium positions are given by $p_i = ih, i = 0, \ldots, N-1$ where the separation $h = l/(N-1)$. The positions at time t are given by $X_i(t) = p_i + x_i(t)$ where $x_i(t)$ represents the displacement of the ith oscillator from its equilibrium position p_i. The force acting on an oscillator i from its neighbor $i+1$ is given by $k(\delta + \alpha\delta^2 + \beta\delta^3)$ where δ is the deviation of their current separation from their equilibrium separation. We will consider the case of a quadratic nonlinearity so that $\beta = 0$ and $\alpha > 0$.. The equations of motion can be expressed as a function of the displacements only so that the equations of motion take the form

$$m\ddot{x}_i = k(x_{i+1} - x_{i-1} - 2x_i)[1 + \alpha(x_{i+1} - x_{i-1})]$$

where the ends of the chain are fixed by boundary conditions $x_0(t) = x_{N-1}(t) = 0$ and the initial conditions have the form $x_i(0) = y_i, \dot{x}_i(0) = 0; i = 1, \ldots, N-2$. In the absence of nonlinear forces, $\alpha = 0$, the system becomes integrable with a solution easily obtained through normal modes $Q_k(t)$ defined by a discrete Fourier representation of the state.

2.3.2 Continuum limit of nonlinear FPU system and soliton behavior

The dynamics in the nonlinear case $\alpha = 0$ can be understood by consideration of the continuum limit $N \to \infty$ which becomes a PDE defining a nonlinear string [73]. To derive this model, let ρ, κ be the density and Young's modulus of the string so that we can rewrite the equations of motion as

$$\ddot{x}_i = c^2 \left(\frac{x_{i+1} + x_{i-1} - 2x_i}{h^2} \right) [1 + \alpha(x_{i+1} - x_{i-1})]$$

where $c = \sqrt{\kappa/\rho}$. We then consider the ODE system as a discretization of a function $u(x,t)$ that measures the displacement from equilibrium at time t of the particle with equilibrium position x. Taylor expanding $u(x,t)$ and keeping terms up to $O(h^2)$ our asymptotic continuum approximation for finite h reads $((1/c^2)u_{tt}) - u_{xx} = (2\alpha h)u_x u_{xx} + \left(\frac{h^2}{12}\right)(h^2/12)u_{xxxx}$. After a careful change of variables and taking the limit as α and h tend to zero at the same rate we obtain

$$y_{\xi\tau} + y_\xi y_{\xi\xi} + \delta^2 y_{\xi\xi\xi\xi} = 0$$

where $\delta = \lim_{h\to 0} \sqrt{h/24\alpha}$. Soliton solutions take the form of a localized pulse

$$y_\xi(\xi, \tau) = \frac{a}{2} \operatorname{sech}^2 \left(\frac{\sqrt{6a}\delta}{2} (\xi - 6a\tau - c_0') \right)$$

whose wave speed $c(1 + 6a\alpha h)$ can be found by transforming back to the original spatial and temporal coordinates x, t.

2.3.3 Compression performance of linear and nonlinear dimensionality reduction methods

The nonlinear model reduction procedure described in section 2.2 uses encoding and decoding transformations to achieve an initial compression of the data into dominant modes. These transformations can be taken from a dimensionality reduction procedure such as PCA where the encoder and decoder are linear

transformations. To find the dimensionality reduction technique that provides the best compression, we will first compare the performance of several linear and nonlinear methods on data generated from the FPU system.

2.3.3.1 Wavelet transform

The first linear dimensionality reduction approach considered is based on the discrete wavelet transform of time series data [46, 91]. The wavelet transform is a linear transformation on the space of functions which provides an efficient basis to capture local features of wave shape. As the soliton solutions of the FPU system take the form of a traveling wave, wavelets are an appropriate choice of basis with the potential to concentrate information in a small number of dominant coefficients.

2.3.3.2 POD

The second linear dimensionality reduction method is taken to be POD [81] which, in the discrete case, is equivalent to Principal Component Analysis (PCA) [48]. This was described in detail in section 1.1.1.2.

2.3.3.3 Autoencoder

By comparison to the two linear encoders considered above, nonlinear encoders realized through neural networks offer the potential for higher compression rates. AE neural networks, introduced in section 1.2.1.2, are the natural choice for nonlinear encoding aspect of the model reduction procedure presented in section 2.2. With a single linear layer forming the encoder and decoder, an AE carries out POD [53] and thus provides a natural nonlinear generalization of POD. To facilitate SGD convergence to effective encoding and decoding transformations, the AE neural network is initialized based on linear POD and guided via a homotopy to a fully nonlinear transformation. The functional form of the AE is given by

$$\boldsymbol{A}(\mathbf{x}, \mathcal{W}) = \mathbf{L}_k^d \circ \phi \circ \cdots \circ \phi \circ \mathbf{L}_1^d \circ \phi \circ \mathbf{L}_k^e \circ \cdots \circ \phi \circ \mathbf{L}_1^e(\mathbf{x})$$

where the $\mathbf{L}_i^e$, $\mathbf{L}_i^d$ are affine transformations whose entries comprise the weights $\mathcal{W}$, ϕ represents a nonlinear elementwise function, and the encoder $\boldsymbol{e}_{\mathcal{W}}$ and decoder $\boldsymbol{d}_{\mathcal{W}}$ are formed by $\phi \circ \mathbf{L}_k^e \circ \cdots \circ \phi \circ \mathbf{L}_1^e$ and $\mathbf{L}_k^d \circ \phi \circ \cdots \circ \phi \circ \mathbf{L}_1^d$, respectively. The dimension is reduced from D to d in a series of steps of size s such that $d = D - ks$ and $\mathbf{L}_i^e : \mathbb{R}^{D-(i-1)s} \to \mathbb{R}^{D-is}$, $\mathbf{L}_i^d : \mathbb{R}^{d+(i-1)s} \to \mathbb{R}^{d+is}$ for $i = 1, \ldots, k$. Given $\mathbf{U}_d$ provides

the reduced basis from POD, the initial form of the AE should satisfy

$$\begin{aligned} \mathbf{L}_k^e \cdots \mathbf{L}_1^e &= \mathbf{U}_d^T \\ \mathbf{L}_k^d \cdots \mathbf{L}_1^d &= \mathbf{U}_d \end{aligned}$$

A natural solution is to define $\mathbf{I}_i^j$ to be the first $j < i$ rows of the i-dimensional identity matrix and form

$$\begin{aligned} \mathbf{L}_k^e \cdots \mathbf{L}_1^e &= \mathbf{I}_{D-(k-1)s}^{D-ks} \cdots \mathbf{U}_{D-s} \\ \mathbf{L}_k^d \cdots \mathbf{L}_1^d &= \mathbf{U}_{D-s}^T \cdots (\mathbf{I}_{D-(k-1)s}^{D-ks})^T \end{aligned}$$

which provides the initial setting of the weights for SGD. To guide this nonlinear regression function to an effective nonlinear generalization of POD, a homotopy is carried out such that the nonlinear elementwise layer ϕ is continuously changed from the linear identity function to ReLu. This homotopy can be expressed as a parametric function

$$\phi(x, \alpha) = (1 - \alpha)x + \alpha \text{ReLu}(x)$$

such that $\phi(x, 0) = x$ and $\phi(x, 1) = \text{ReLu}(x)$. The parameter α controls the nonlinearity of the function and is incrementally increased from 0 to 1 during SGD to provide an approximately smooth transition into a fully nonlinear DNN. This procedure guides SGD to a significantly better encoding, decoding pair than the standard approach of randomly initializing the weights of the network.

2.3.3.4 Results

First, the three dimensionality reduction methods were evaluated on a random sample of snapshots from the trajectory of the FPU system with quadratic nonlinear corresponding to the continuum limit described in section 2.3.2. The mean squared reconstruction error relative to the domain length was computed for each method across a set of reduced dimensions to extract compression performance as a function of dimension.

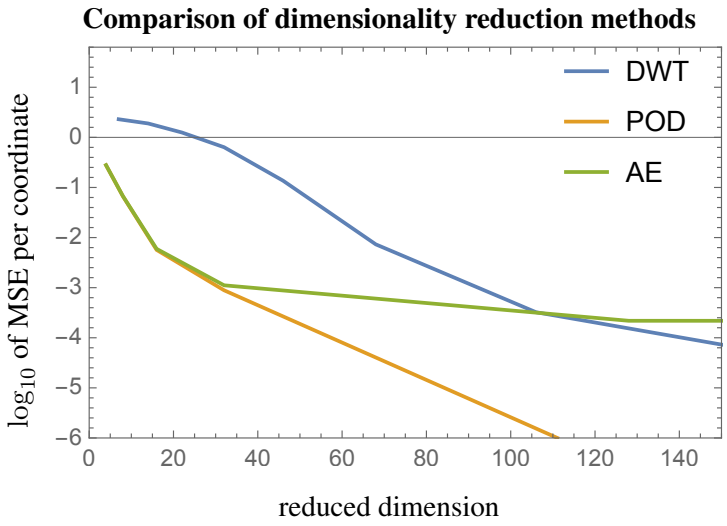


Figure 2.2: Dimensionality reduction performance of DWT, POD, and AE where error is computed with respect to string length.

The nonlinear compression carried out by the AE performs poorly for higher dimensions but coincides with POD as the dimension decreases. The improvement in performance for smaller dimensions reflects the higher compression ratio potential of nonlinear dimension reduction. The AE is not able to exceed the compression of POD in this case because the simplicity of the nonlinear spring dynamics yields data that lies sufficiently close to a linear subspace. On the other hand, the wavelet transform exhibits reconstruction error approximately three orders of magnitude larger than for POD across the dimensions sampled. While only a narrow band of wave scales are needed to represent the temporally consistent shape of the soliton, a significant number of shift parameters are needed to capture the translation of the wave across the entire domain which limits the compression potential.

2.3.4 Extracting soliton behavior via nonlinear model reduction procedure

To test the model reduction procedure, the FPU system was numerically integrated with initial conditions given by correlated excitation of the oscillators with the hyperbolic secant function defining the soliton wave profile. The initial pulse divides into a leftward and rightward traveling wave of half the amplitude. These solitons reflect across the horizontal axis at the boundaries of the domain and pass through each other with minimal change in shape. The DNN model reduction procedure was carried out to obtain an approximate mesoscale forward map which is validated by verifying it captures the correct behavior of the soliton solution.

2.3.4.1 Robustness of learned mesoscale forward map

The encoder $\boldsymbol{E} : \mathbb{R}^D \to \mathbb{R}^d$ and decoder $\boldsymbol{D} : \mathbb{R}^d \to \mathbb{R}^D$ are taken to be $\mathbf{U}_d^T \in \mathbb{R}^{d \times D}$ and $\mathbf{U}_d \in \mathbb{R}^{D \times d}$, respectively, where $\mathbf{U}_d$ is unitary with d columns determined by POD which performed significantly better than the AE and DWT encoders. This provides the dimension reduction component of the acceleration procedure which can now be evaluated on the FPU system by constructing the reduced mesoscale forward map. Following the model reduction approach described in section 2.2, a forward map was trained to carry out forward integration of a mesoscale defined by a mesostep $\tau = \frac{T_{\text{sol}}}{10}$ where T_{sol} is taken to be the time for one traversal of the soliton across the domain. The behavior of a typical trained forward map is show in figure 2.3.

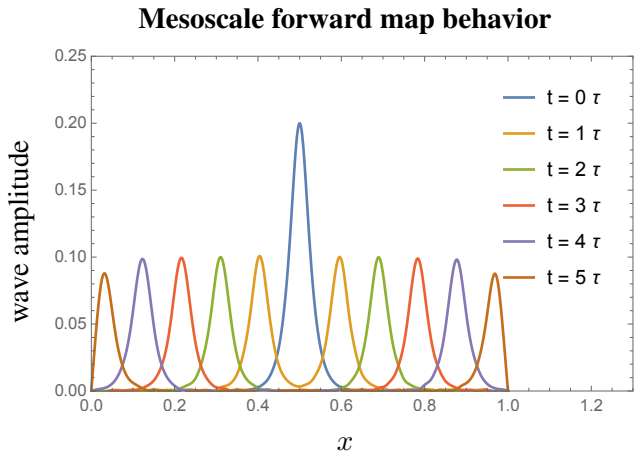


Figure 2.3: Repeated application of learned mesoscale forward map on the FPU initial condition.

Repeated application of the forward map approximates integration of the system forward in time across mesosteps τ. The learned mesoscale forward map captures the correct shape and movement of the wave and the compounding of local errors is sufficiently slow to allow for the advancement of the system across a significant number of mesosteps. The growth of the relative L^2 error from iterative application of the forward map is shown in figure 2.4.

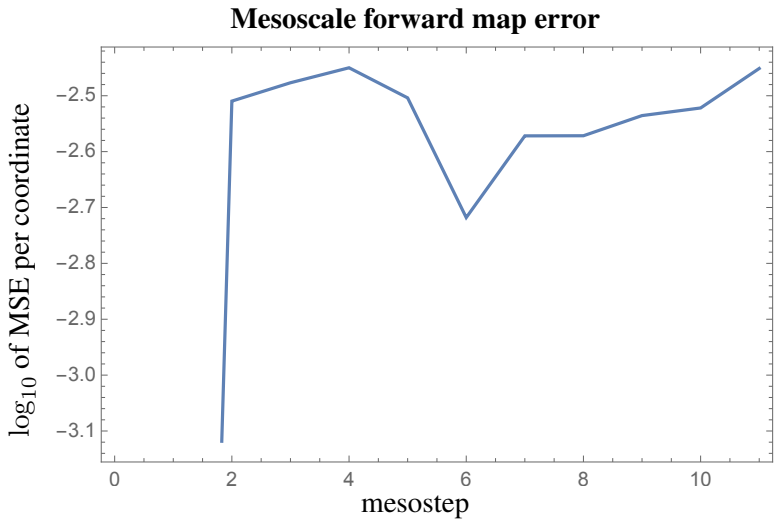


Figure 2.4: L^2 error growth during iterative application of the learned forward map

2.3.4.2 Extraction of correct wave speed

To verify that the forward map captures the correct wavespeed, the movement of the soliton waves under the forward map was tracked and used to compute an approximate wavespeed as a function of the length of the nonlinear string. This extracted wavespeed is compared to the analytical wavespeed in figure 2.5.

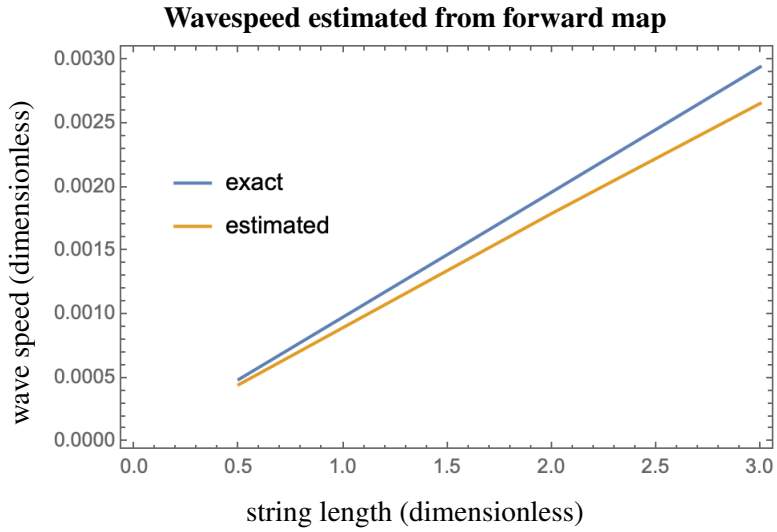


Figure 2.5: Comparison of exact soliton wavespeed with estimation from the mesoscale forward map.

2.4: APPLICATION TO MD

In this section we apply the nonlinear model reduction procedure from 2.2 to MD simulations of a simple protein undergoing alpha helix nucleation. The system is governed by a highly nonlinear potential and driven by a stochastic heat bath that approximately maintains a fixed temperature. While the current model reduction procedure under consider does not account for stochastic behavior, it is instructive to verify it can learn the average slow modulation of the folding protein dynamics.

2.4.1 MD theory

The MD simulation is governed by the stochastic differential equation

$$\begin{bmatrix} \mathrm{d}\boldsymbol{x}_t \\ \mathrm{d}\boldsymbol{p}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{M}^{-1}\boldsymbol{p}_t \\ \mathbf{0} \end{bmatrix} \mathrm{d}t + \begin{bmatrix} \mathbf{0} \\ -\nabla_{\boldsymbol{x}_t} U(\boldsymbol{x}_t) \end{bmatrix} \mathrm{d}t + [\text{thermostat}] \tag{2.6}$$

where the thermostat term accounts for the random forcing of the system due to interactions with unresolved solvent molecules. The potential in (2.6) takes the form

$$\begin{aligned} U(\mathbf{x}) &= \sum_{\text{bonds}} k^b_{ij}(r_{ij}(\mathbf{x}) - r_0)^2 \\ &+ \sum_{\text{angles}} k^a_{ijk}(\theta_{ijk}(\mathbf{x}) - \theta_0)^2 \\ &+ \sum_{\text{torsions}} k^t_{ijkl}[1 + \cos(n_{ijkl}\phi_{ijkl}(\mathbf{x}) - \phi_0)] \\ &+ \sum_{j=1}^{N-1}\sum_{i=j+1}^{N} f_{ij}\left\{\epsilon_{ij}\left[\left(\frac{r^0_{ij}}{r_{ij}(\mathbf{x})}\right)^{12} - 2\left(\frac{r^0_{ij}}{r_{ij}(\mathbf{x})}\right)^{6}\right] + \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}(\mathbf{x})}\right\} \end{aligned}$$

where $r_{ij}, \theta_{ijk}, \phi_{ijkl}$ give the distance between atoms i, j, angle between atoms i, j, k and dihedral angle of atoms i, j, k, l, respectively and r_0, θ_0, ϕ_0 their equilibrium values. The constants $k^b_{ij}, k^a_{ijk}, k^t_{ijkl}$ define the magnitude of the restoring forces resulting from perturbation from equilibrium values of the bond, angle, and torsion terms, respectively, which q_i represents the electric charge of atom i [76, 29]. As a simple example of protein folding, we consider protein consisting of Alanine amino acids. Alanine has a strong tendency for alpha helix formation so that we can expect to observe some coil formation on the timescale sampled by MD [34, 93].

2.4.2 Setup and workflow of MD simulations

MD simulations are complex and time-consuming to carry out involving many physical quantities that must be kept track of for validation and testing. To set up and run the simulations, an extensive workflow was created as part of the contributions in this thesis. The first aspect of this workflow consists of extracting protein data and AMBER potential parameters from files such as the initial geometry of the molecule, collections of atoms involved in bonds and torsion arrangements, and the various force constants defining these terms based on the types of atoms involved. These were all extracted such that the AMBER potential function could be reconstructed independently and used to validate simulations.

The second aspect of the workflow involved extensive code to run the MD simulations automatically. These simulations generate extremely large amounts of data over short periods of simulation time on the order of tens of megabytes per picosecond. This substantial output of data coupled with the complex mesoscale sampling procedure outlined in section 2.2 necessitated the development of a computational framework for repeatedly invoking the MD simulation to generate each mesostep separately. Given the initial parameters of the simulation which define the microscale, mesoscale, simulation time, and molecular parameter files, the computational framework automatically generates a bash script containing a large sequence of terminal commands that call the MD simulation program in the appropriate pattern. This allows for complex mesoscale datasets to be generated efficiently with only the adjustment of a few parameters.

The final aspect of the workflow consists of automatically extracting data from the large collection of files generated by MD which contain atomic configurations separated into mesosteps. This data is extracted and processed and used to set up the training procedures for the various DNNs used for model reduction.

To develop a specific MD data set for the applications of the methods presented in 2.2 a protein was constructed by connecting 20 Alanine amino acids in sequence. MD simulation of the chain was carried out using the workflow described above over 10 ns with an initially stretched, linear configuration of the protein.

2.4.3 Results

2.4.3.1 Comparison of linear and nonlinear dimensionality reduction

We would like to compare the performance of the model reduction procedure in the cases where the forward map uses linear and nonlinear encoding onto a reduced space of dominant modes. POD will be

used for the linear procedure while an AE will be used to carry out a nonlinear encoding. The compression performance of these two approaches on a random sample of MD data is depicted in figure 2.6.

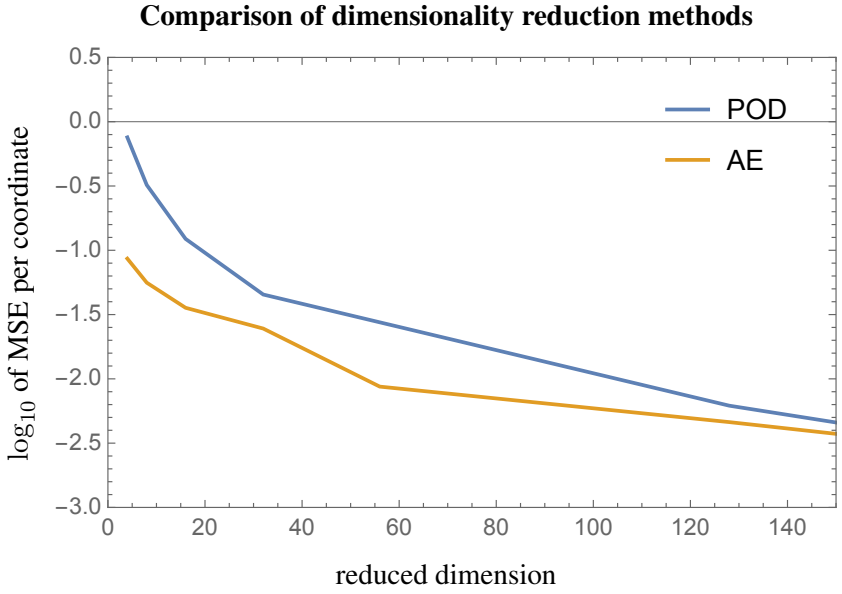


Figure 2.6: Dimensionality reduction performance of DWT, POD, and AE where error is computed with respect to protein length.

The AE provides better compression across the range of dimensions considered with the relative performance improving as dimension decreases. At the lowest dimension considered, the AE provides an order of magnitude lower reconstruction error suggesting that the MD trajectories indeed lie on a nonlinear submanifold of phase space that is picked up by the training process.

2.4.3.2 Learned forward map

A mesostep $\tau = \frac{T_{\text{helix}}}{1000}$ is introduced to define a mesoscale for the DNN acceleration procedure where T_{helix} is approximately the timescale for alpha helix nucleation. The mesoscale forward map for both the POD encoder and AE was then approximated through deep neural network regression on data sampled from a single MD simulation. Initially, we would like to evaluate the performance of this approximate forward integrator over the time interval where training took place. The behavior of the approximate forward map is depicted in figure 2.7 where both the entire predicted configuration of atoms is visualized along with the backbone of the molecule only. The backbone of the molecule consists of carbon and nitrogen atoms bonded in a linear sequence and reflects the overall geometry of the molecule. In this case, a reduced dimension

$d = 60$ was chosen as this captures 99.9% of the energy in the sense of POD modes. As the positions predicted by the POD forward map and the AE forward map appear quite similar, only the positions given by the POD forward map are shown in 2.7 .

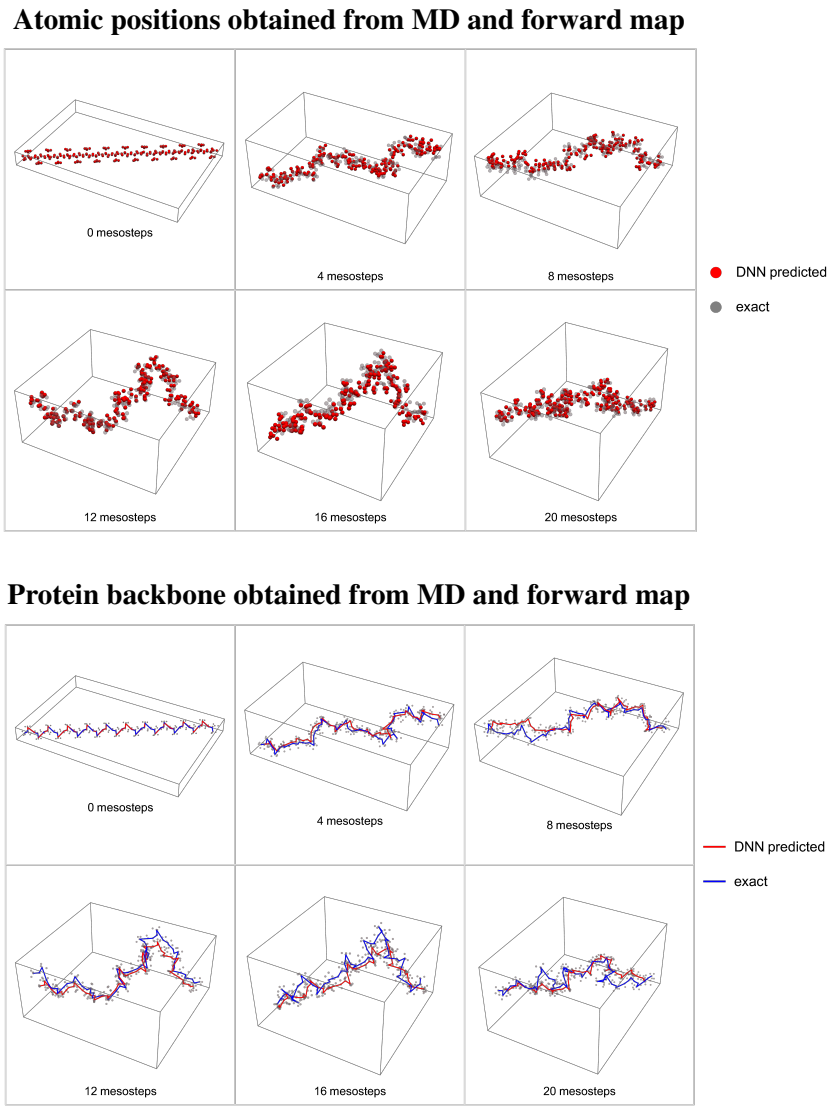


Figure 2.7: Atomic positions (top) and backbone positions (bottom) predicted by the learned forward map using POD encoding.

To quantify the performance of the linearly and nonlinearly encoded forward maps, the maps were iteratively applied to a starting configuration and the predicted positions compared with respect to the exact positions generated by DNS. The error measure $e(\mathbf{x}, \hat{\mathbf{x}})$ between exact coordinates $\mathbf{x}$ and predicted

coordinates $\hat{\mathbf{x}}$ is introduced and defined by

$$e(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{aD} \sum_{i=1}^{D} ([\mathbf{x}]_i - [\hat{\mathbf{x}}]_i)^2 \tag{2.7}$$

where a is the average interatomic spacing among the equilibrium bond lengths across the bonds present in the molecule. The represents the mean squared reconstruction error (MSE) per coordinate relative to the characteristic length scale a. As the atoms undergo thermally driven oscillations whose maximal amplitude is bounded above by bond length, this length scale represents a degree of positional uncertainty inherent in any prediction since we are neglecting the microscale thermal modes of the system.

The relative error e was computed as a function of the number of application of the mesoscale forward map in both the AE and POD cases. This was done across a range of reduced dimensions d as this reflects one of the central parameters of the model reduction procedure. These errors were averaged by computing an ensemble of predicted configurations starting from various exact initial conditions. This is shown in figure 2.8.

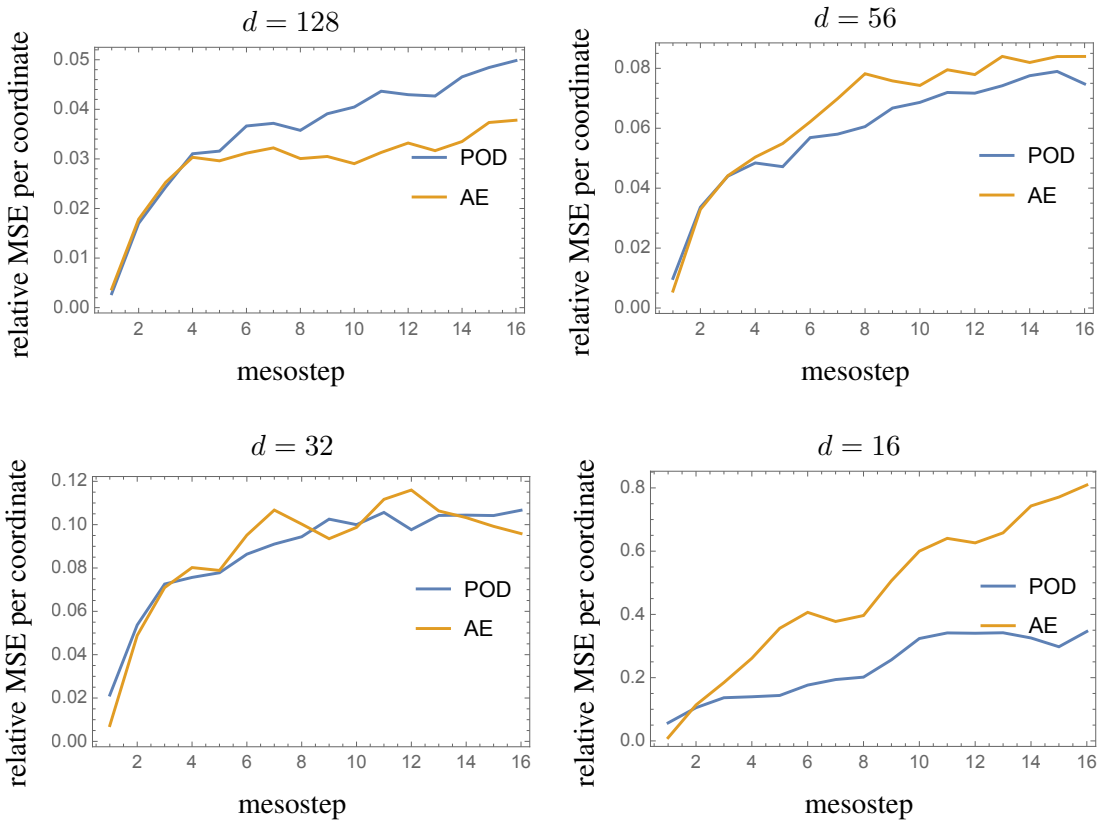


Figure 2.8: Mean squared relative error of configurations predicted by the POD and AE forward maps across several values of d.

The AE forward map shows slower error growth at higher dimensions $d = 128$ but becomes less stable over iterations as d decreases. This suggests the AE is providing better reconstruction of the data but that DNNs are not able to approximate the nonlinearly encoded forward map as the complexity of the network decreases. Both the POD and AE forward show slow error growth at sufficiently large values of d indicating that the procedure finds an accurate, reduced approximation of the forward map over the time interval defined by the training data.

Note that approximate forward integration of the MD system over this time interval provides no acceleration because we are not sampling configurations beyond what has already been generated by MD. We would like the forward map to extrapolate the positions beyond the training data and obtain reasonable predictions computed at reduced computational cost compared to DNS. In figure 2.9, the error of extrapolation beyond

the training data using the forward is plotted again as a function of iteration for both the POD and AE forward maps and depicted for the same values of d.

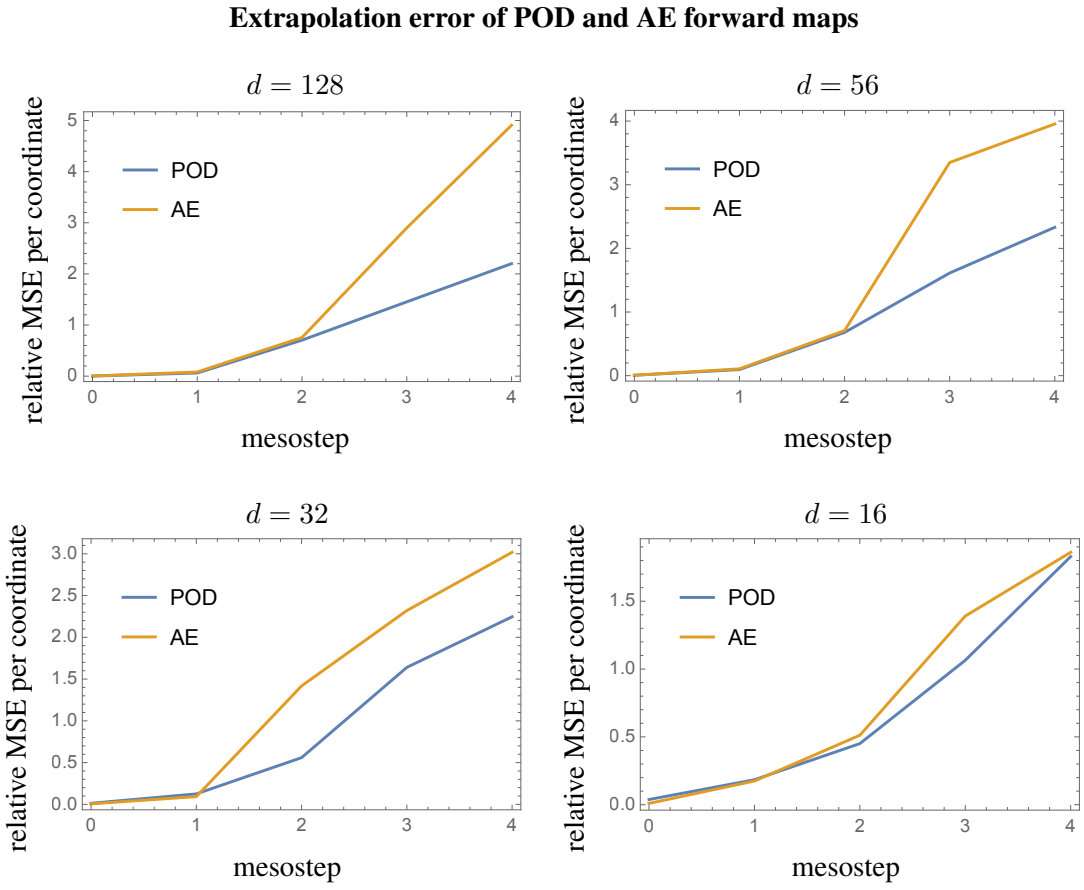


Figure 2.9: Extrapolation error of POD and AE forward maps across several values of d

The extrapolation procedures predict atomic positions to within the average interatomic spacing a, for 2 to 3 mesosteps depending on the reduced dimension d. The mesostep consists of $6 \cdot 10^5$ microscale time steps such that advancement of the configuration by 3 mesosteps represents a significant acceleration of the DNS. The time interval required to carry out 3 mesosteps of DNS is on the order of three hours whereas this model reduction procedure can be carried out in less than 10 minutes resulting in roughly a twenty-fold increase in speed.

Note that as the d decreases, we see a significant improvement in extrapolation performance with both methods. Decreasing the reduced dimension forces the procedure to approximate the state of the system and its mesoscale dynamics with only a small number of the most dominant modes. The slower growth in extrapolation error indicates the evolution of these modes is captured by the forward map for a longer period

of time into the future than with the less dominant modes. This is unsurprising as we would expect the large scale modes of the system to evolve over a longer timescale than the smaller, more thermally driven modes.

2.4.3.3 Robustness to perturbations

Another important feature captured by the learned forward map is robustness to perturbations of the initial data. Proteins can take different pathways to the final folded state and we would like the forward map to produce configurations with similar backbone structure despite starting from a perturbed initial condition. The initial linear chain was subjected to a transformation inducing curvature along the molecule in one direction as a physically realistic modification. The relative error e was plotted as a function of iteration for several different values of the initial curvature modification.

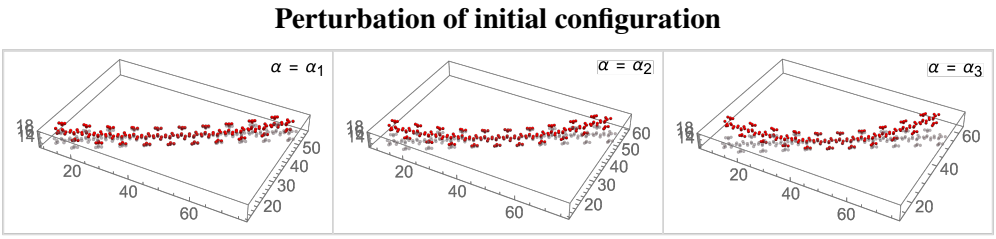


Figure 2.10: Perturbation of initial linear protein configuration.

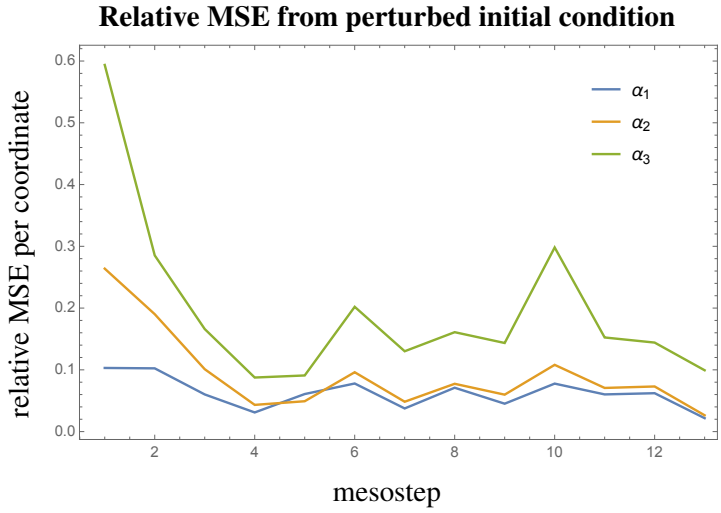


Figure 2.11: Relative MSE of perturbed configuration as a function of forward map iteration.

Observe that each perturbed configuration starts with a large relative error but converges back toward the exact solution as the forward map is repeatedly applied. This shows the learned forward map is robust to physically realistic perturbations of the data.

2.5: CONCLUSION

Deep neural networks combined with nonlinear dimensionality reduction methods are able to learn an efficient, approximate forward map that can advance a multiscale nonlinear system at the mesoscale. The forward map displays robustness to perturbations and to compounding error when applied over the training interval. It can also be used to extrapolate the state of the system beyond the data generated by DNS for a small number of mesostep intervals. This creates the potential for acceleration of MD by periodically advancing the system at the mesoscale using the learned forward map which captures the slow modulation of the protein.

CHAPTER 3: NONLINEAR STOCHASTIC MODEL REDUCTION THROUGH ACCELERATION OF SGD

3.1: INTRODUCTION

The previous chapter concluded that functions defined by DNNs are a suitable basis for nonlinear model reduction procedures. While AEs provide a natural architecture for the nonlinear encoding of data into spaces of smaller dimension, the approach described in this thesis attempts to extend the applicability of DNNs to systems which are both dynamic and stochastic. To that end, consider a stochastic differential equation form

$$\mathrm{d}\boldsymbol{x}_t = \boldsymbol{V}(\boldsymbol{x}_t)\mathrm{d}t + \mathrm{d}\boldsymbol{B} \tag{3.1}$$

where $\boldsymbol{x}_t(\omega) \in \mathbb{R}^D$, with D large, and $\boldsymbol{V}(\boldsymbol{x})$ nonlinear. We would like to use DNNs to capture the slow, dominant modes of the system which reflect the dynamics over large timescales.

The forward operator $\mathcal{F}_{t_1,t_2}$ for 2.2 is time-invariant such that $\mathcal{F}_{t_1,t_2} = \mathcal{F}_{t_1',t_2'}$ as long as $t_2 - t_1 = t_2' - t_1'$ and can be written as $\mathcal{F}_{\Delta t}$, a function of time difference only. The corresponding forward operator for 3.1 modulates over time such that a temporal sequence $\boldsymbol{F}_i(\mathbf{x}, \boldsymbol{\mathcal{W}}_i)$ of DNN approximations is required to capture the changing modes of the system. The weights $\boldsymbol{\mathcal{W}}_i$ are assumed to reflect the dominant modes of 3.1 similar to the matrix entries of $\mathcal{F}_{\Delta t}$ reflecting the dominant modes of 2.2.

It remains to consider how to account for the stochastic component of 3.1. Recall that the Karhunen-Loève expansion [63, 56] of the stochastic process $\boldsymbol{x}_t$, $t \in [a, b]$ has the form

$$\boldsymbol{x}_t(\omega) = \sum_{k=1}^{\infty} Z_k(\omega) e_k(t)$$

where the Z_k are pairwise uncorrelated random coefficients of the deterministic modes $e_k(t)$. Truncating this to the first d terms provides the reduced representation of the stochastic process $\boldsymbol{x}_t(\omega) = Z_1(\omega)e_1(t) + \cdots + Z_d(\omega)e_d(t)$. Similar to the random coefficients $Z_k(\omega)$, the sequence of weights $\boldsymbol{\mathcal{W}}_i$ inherits stochastic behavior from the randomness inherent in the SDE which suggests viewing the weights $\boldsymbol{\mathcal{W}}_i$ as samples

of a vector-valued stochastic process $\boldsymbol{w}_t(\omega)$ that provide reduced stochastic coefficients representing the original process. This can be thought of as a rudimentary attempt to obtain a nonlinear generalization of a Karhunen-Loève expansion.

The ultimate goal is then to achieve acceleration by first estimating the statistics of this process and sampling consistent future values from the appropriate probability distribution. These estimated future values of the weights could then be used to define a new DNN forward map used to advance the system into the future at the mesoscale. Defining a physically realistic forward map through stochastic process extrapolation is a difficult task, hence we will instead investigate the more modest goal of obtaining a reasonably accurate estimate of the evolution of the forward map where accuracy is measured by acceleration of SGD.

The remainder of this chapter presents the details of the SGD acceleration procedure. As discussed above, sequences of DNNs are trained to approximate the forward propagation map such that the weights can be viewed as representing a kind of nonlinear mode of the system. We adopt the perspective of considering sequences of weights as stochastic processes induced by the underlying stochasticity of the microscale model. The statistical behavior of the resulting stochastic processes is of particular interest and several different statistical assumptions are investigated and compared. Acceleration of SGD for the next forward map in the sequence is carried out through sampling the underlying probability distributions for the weights to generate a subsequent DNN that can be used to start SGD closer to a local minimum that choosing a random starting point.

3.2: STOCHASTIC MODEL REDUCTION PROCEDURE

This section presents the details of the SGD acceleration procedure. Sequences of DNNs are trained to approximate the forward map such that the weights can be viewed as representing a kind of nonlinear, stochastic mode of the system which evolves at the mesoscale. We adopt the perspective of considering sequences of weights as stochastic processes induced by the underlying stochasticity of the microscale model. SGD acceleration is carried out by extrapolation of the stochastic processes defined by the weights. This requires modeling the temporally varying joint distribution governing the processes and extrapolation of its moments. Several possible approaches will be discussed.

Deep neural networks provide the nonlinear transformation that determines the temporal sequence of stochastic reduced coordinates defining the mesoscale reduced system. The reduced coordinates are taken

to be the weights of the DNNs so that the initial stage of SGD acceleration consists of training a sequence of DNN to carry out the mesoscale forward map and then extracting the time series of weights. Consider an SDE of the form (3.1) with exact solution $\boldsymbol{x}_t(\omega)$ approximated by direct numerical simulation with time step Δt chosen based on the fast dynamics of the system. The time step Δt defines the microscale of the problem and is assumed to be small with respect to the mesoscale $\Delta\tau = N_{\text{meso}}\Delta t$ on which the large scale behavior of interest takes place. The resulting sequence of positions is given by $\boldsymbol{x}(t_i)$ where $t_i = i(\Delta t)$. DNN regression is carried out to approximate the forward map which advances the state one mesostep into the future. Letting $\tau_i = i\Delta\tau$, the input data for training can be expressed as the set of atom positions $\{\mathbf{x}_m = \mathbf{x}(\tau_M + t_m)\}_{m=1}^{N_{\text{meso}}}$ following mesostep $\tau_M = M\Delta\tau$ and the output data is given by atom positions one mesostep later $\{\boldsymbol{y}_m = \mathbf{x}(\tau_{M+1} + t_m)\}_{m=1}^{N_{\text{meso}}}$ where $M = 1, \ldots, N_\tau$. The training process to determine weights $\boldsymbol{\mathcal{W}}_M$ for the forward map approximant $\boldsymbol{F}_M(\mathbf{x}, \boldsymbol{\mathcal{W}}_M)$ is carried out through stochastic gradient descent $\boldsymbol{\mathcal{W}}_M^{n+1} = \boldsymbol{\mathcal{W}}_M^n - \eta\nabla Q(\boldsymbol{\mathcal{W}}_M^n)$ where Q is a mean squared error cost function over the training data. Repeating this regression procedure across a sequence of mesosteps results in an associated sequence of DNN functions

$$\boldsymbol{F}_M(\mathbf{x}, \boldsymbol{\mathcal{W}}_M) : \mathbb{R}^D \times \mathbb{R}^{D_w} \to \mathbb{R}^D, M = 1, \ldots, N_{\text{meso}}$$

where $\boldsymbol{\mathcal{W}}_M \in \mathbb{R}^{D_w}$ for each M and $\boldsymbol{F}_M(\mathbf{x}, \boldsymbol{\mathcal{W}}_M)$ is given by a standard FFNN of the form (1.7). This process of generating a sequence of temporally varying forward maps is depicted in figure 3.1.

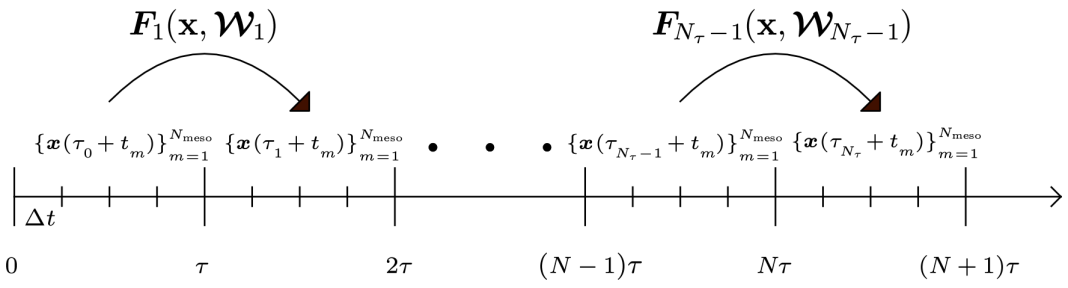


Figure 3.1: Sequence of DNNs generated for acceleration of SGD.

The time series of weights $\boldsymbol{\mathcal{W}}_1, \ldots, \boldsymbol{\mathcal{W}}_{N_\tau-1}$ is viewed as a sample from some vector-valued stochastic processes $\boldsymbol{w}_\tau(\omega)$ at the mesoscale τ defined by its finite-dimensional distributions $p(\boldsymbol{w}_n, \tau_n; \ldots; \boldsymbol{w}_1, \tau_1) = P\{\boldsymbol{w}_{\tau_n}(\omega) = \boldsymbol{w}_n, \ldots, \boldsymbol{w}_{\tau_1}(\omega) = \boldsymbol{w}_1\}$. Acceleration is achieved by estimation of these distributions,

extrapolating their moments, and sampling these new distributions to obtain an estimate of the DNN weights at the future time step, $\hat{\boldsymbol{\mathcal{W}}}_{N_\tau}$.

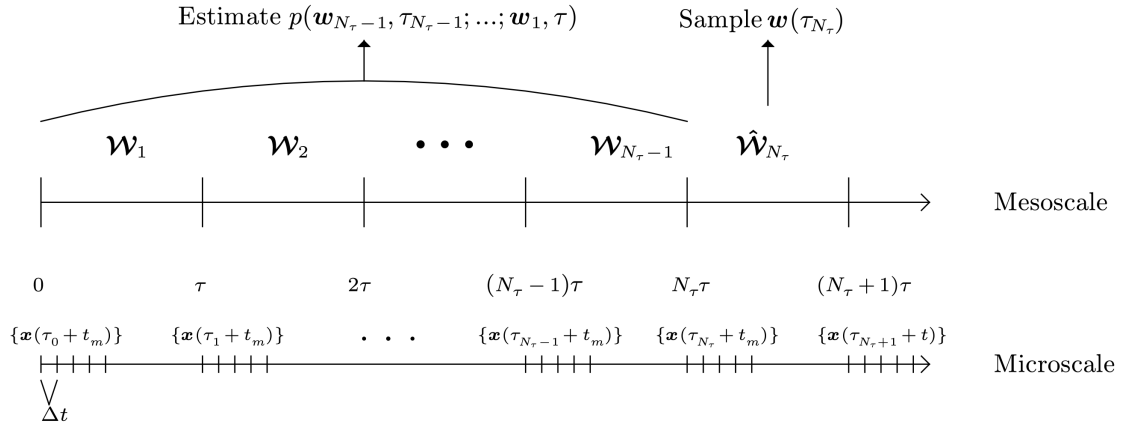


Figure 3.2: Extrapolation of the stochastic sequence of weights.

The above procedure can be summarized in the following algorithm which returns an estimate of the next value of the collection of weights given by $\hat{\boldsymbol{\mathcal{W}}}_{N_\tau}$. This estimate is then used as an initial value for training the next forward map through SGD.

Algorithm 1: SGD acceleration procedure

Generate system snapshots $\boldsymbol{x}(t_i)$ via DNS;
for $i = 1$ *to* $N_{\tau-1}$ **do**
  Form training data $\{\mathbf{x}_m = \mathbf{x}(\tau_M + t_m)\}_{m=1}^{N_{\text{meso}}}$, $\{\boldsymbol{y}_m = \mathbf{x}(\tau_{M+1} + t_m)\}_{m=1}^{N_{\text{meso}}}$;
  Train ith forward map $\boldsymbol{F}_i(\mathbf{x}, \boldsymbol{\mathcal{W}}_i)$ via SGD ;
end
Extract weight sequence $\boldsymbol{\mathcal{W}}_1, \ldots, \boldsymbol{\mathcal{W}}_{N_\tau-1}$;
Estimate $p(\boldsymbol{\mathcal{W}}_n \tau_n; \ldots; \boldsymbol{\mathcal{W}}_1, \tau_1)$;
Result: $\hat{\boldsymbol{\mathcal{W}}}_{N_\tau}$ estimated through sampling PDF

Figure 3.3: SGD acceleration algorithm.

It remains to consider what type of stochastic process the weights may represent. This amounts to assuming a particular form of joint distribution over finite samples from the stochastic process. A relevant property of DNNs is that, in the limit of infinite breadth, a DNN converges to a Gaussian distribution over the space of functions, i.e., a Gaussian process [61]. Networks with Gaussian weights have also been shown to perform a distance-preserving embedding of the data into feature space [40]. Hence, modeling the weights as Gaussian processes is a natural starting point.

3.2.1 Modeling weights as Gaussian stochastic processes

The random forcing terms present in MD motivates viewing the sequences of weights as stochastic processes. More precisely, weights from a specific layer are considered as samples from a vector-valued random process $\boldsymbol{w}_\tau(\omega) \in \mathbb{R}^d$ defined by the statistics of their finite dimensional distributions. As part of the procedure presented here, we consider several different assumptions about the form of these distributions. The discussion preceding this section motivates our initial approach of modeling the $\boldsymbol{w}_\tau(\omega)$ as Gaussian processes. To avoid consideration of large kernels for vector-valued processes, we instead assume the weights are locally Gaussian in the sense that $\boldsymbol{w}_\tau(\omega) \sim \mathcal{N}(\boldsymbol{\mu}_\tau, \boldsymbol{\Sigma}_\tau)$ where $\boldsymbol{\mu}_\tau = \boldsymbol{\mu}(\tau), \boldsymbol{\Sigma}_\tau = \boldsymbol{\Sigma}(\tau)$ are functions defining the variation of the mean and covariance.

Suppose we have N_M samples of the weights $\boldsymbol{w}_{\tau_1}, \ldots, \boldsymbol{w}_{\tau_{N_M}}$ obtained from procedure in section 3.2 that define a sequence of DNNs over N_M mesosteps. Construct data matrices

$$\mathbf{X}_{\tau_i} = [\boldsymbol{w}_{\tau_i} \boldsymbol{w}_{\tau_{i+1}} \cdots \boldsymbol{w}_{\tau_{i+k-1}}]$$

taken from overlapping windows of the weights. The locally Gaussian assumption then takes the form

$$\mathbf{X}_{\tau_i} = [\boldsymbol{w}_{\tau_i}(\omega_1) \cdots \boldsymbol{w}_{\tau_i}(\omega_k)], \boldsymbol{w}_{\tau_i} \sim \boldsymbol{\mathcal{N}}(\boldsymbol{\mu}_{\tau_i}, \boldsymbol{\Sigma}_{\tau_i})$$

By extrapolating the time-series of moments $\boldsymbol{\mu}_{\tau_i}, \boldsymbol{\Sigma}_{\tau_i}$, we can obtain weights for future mesostep $\tau_{N_{M+1}}$ that can accelerate the convergence of gradient descent for the next mesostep.

Extrapolation of the covariance matrix The first moments $\boldsymbol{\mu}_\tau$ provide the underlying deterministic modulation of the stochastic process with noise being superimposed according to the covariance. Extrapolation of the means $\boldsymbol{\mu}_\tau$ can be carried out through sampling of an interpolating function outside of the interpolation domain. The covariance $\boldsymbol{\Sigma}_\tau$ describes the temporal behavior of the noise, and interpolation procedures should conserve the symmetric, positive definite property of the matrix. One approach is to interpolate on the tangent space of the manifold of symmetric, positive-definite matrices using the exponential map [103, 7, 9]. A more efficient solution is available when the eigenmodes of the covariance matrix, denoted $\mathbf{u}_\tau^1, \ldots, \mathbf{u}_\tau^d$, exhibit slow variation with respect to the mesoscale τ. Let $\boldsymbol{\Sigma}_{\tau_1}, \ldots, \boldsymbol{\Sigma}_{\tau_n}$ be a sequence of covariance matrices and consider the unitary diagonalizations $\mathbf{U}_{\tau_1} \boldsymbol{\Lambda}_{\tau_1} \mathbf{U}_{\tau_1}^T, \ldots, \mathbf{U}_{\tau_n} \boldsymbol{\Lambda}_{\tau_n} \mathbf{U}_{\tau_n}^T$. By extrapolating $\boldsymbol{\Lambda}_\tau$ positive, diagonal

from data $\mathbf{\Lambda}_{\tau_1}, \ldots, \mathbf{\Lambda}_{\tau_n}$, we obtain $\mathbf{\Lambda}_{\tau_{n+1}}$ and form $\mathbf{\Sigma}_{\tau_{n+1}} \approx \mathbf{U}_{\tau_n} \mathbf{\Lambda}_{\tau_{n+1}} \mathbf{U}_{\tau_n}^T$ where the eigenmodes have been carried over from the previous mesostep.

When the covariance eigenmodes do not exhibit slow variation, we can instead rely on the Cholesky decomposition $\mathbf{\Sigma} = \mathbf{L}\mathbf{D}\mathbf{L}^T$, where $\mathbf{L}$ is unit lower-triangular and $\mathbf{D}$ a positive, diagonal matrix. Given arbitrary $\mathbf{L}$ unit lower-triangular, and positive diagonal matrix $\mathbf{D}$, the product $\mathbf{L}\mathbf{D}\mathbf{L}^T$ will be s.p.d. and a valid covariance matrix. Again letting $\mathbf{\Sigma}_{t_1}, \ldots, \mathbf{\Sigma}_{t_n}$ be a sampling of the trajectory of $\mathbf{\Sigma}_t$, we can extrapolate the covariance matrices through sampling of an interpolating function. Interpolation is straightforward using Cholesky decompositions $\mathbf{L}_{\tau_1}\mathbf{D}_{\tau_1}\mathbf{L}_{\tau_1}^T, \ldots, \mathbf{L}_{\tau_n}\mathbf{D}_{\tau_n}\mathbf{L}_{\tau_n}^T$ and observing that we can represent the unit triangular $\mathbf{L}_{\tau_i} \in \mathbb{R}^{r \times r}$ as a vector $\boldsymbol{l}(t_i) \in \mathbb{R}^{r(r-1)/2}$ and $\mathbf{D}_{\tau_i} \in \mathbb{R}^{r \times r}$ as the vector $\boldsymbol{d}(t_i) \in \mathbb{R}^r$ where if $\boldsymbol{D}_{t_i} = \operatorname{diag}(d_1, \ldots, d_r)$ then $\boldsymbol{d}(t_i) = [\ln d_1 \cdots \ln d_r]^T$ so that we enforce positivity of the diagonal entries.

3.2.2 Modeling weights as non-Gaussian stochastic processes

Viewing DNNs as Gaussian processes is typically motivated with a central limit theorem argument where the number of layers or nodes is taken to approach infinity. The DNNs considered in this approach consist of a relatively small number of layers and hidden nodes in order to make analysis of the weights computationally tractable. Hence, the Gaussian assumption is less motivated in this small neural network context. Furthermore, while homogenization theory predicts Gaussian noise in the case of infinite timescale separation, here we consider models with only moderate, mesoscale separation between slow and fast dynamics. This suggests investigating whether stochastic processes sampled by the weights posses non-Gaussian joint distributions.

3.2.2.1 Extrapolation of the mean using least-squares

The theory for non-Gaussian processes is less extensive, suggesting a first approach is to extrapolate only the first moment of the underlying distribution and ignore the higher moments since we do not have a known statistical description. The mean can be extrapolated by choosing basis functions, such as the standard polynomials, and using least squares projection to compute the best fit linear combination with respect to the 2-norm of mean data. By sampling the resulting function outside the domain of regression, we can extrapolate the mean to future mesosteps. Varying the dimension of the basis allows for adjustments to the smoothness of the function approximation.

More precisely, let $\boldsymbol{f}_n \equiv \boldsymbol{f}(\tau_n)$ and define the history dependence parameter τ^* as the length of the time history considered so that the extrapolation to approximate $\boldsymbol{f}_{n+1}$ is computed from least-squares using

points $\boldsymbol{f}_{n-\tau^*+1}, \ldots, \boldsymbol{f}_n$. For the case of the standard polynomial basis $1, x, \ldots, x^d$, define the smoothness parameter σ by $\sigma = \frac{\tau^*-d-1}{\tau^*-2}$ where $d = 1, \ldots, \tau^* - 1$ is the maximal degree of polynomial the basis can represent. Polynomial interpolation corresponds to a smoothness factor of zero as the approximating function $\widehat{\boldsymbol{f}}(\tau)$ agrees with $\boldsymbol{f}(\tau)$ at each point. On the other hand, least squares fit to a linear function has a maximal smoothness factor of one. As a final remark, note that the standard polynomial basis may not be the optimal choice of representation and more appropriate bases can be heuristically determined from examination of the data.

3.2.2.2 Extrapolation using DNNs

The least squares approach to extrapolation fits data to linear combinations of basis functions by minimizing a quadratic error. Since the weight data we would like to extrapolate is extracted from DNNs trained on nonlinear physics, we should not expect linear combinations of functions to capture the behavior of the weights. To expand the class of nonlinear functions representable with the extrapolation procedure, we can consider compositions of nonlinear functions. This naturally leads to using DNNs as an extrapolation procedure.

The mean $\boldsymbol{\mu}_{\tau_{i+1}}$, where $\tau_i = i(\Delta\tau), i \in \mathbb{Z}^+$, correlates with its previous τ^* values $\boldsymbol{\mu}_{\tau_{i-(\tau^*-1)}}, \ldots, \boldsymbol{\mu}_{\tau_i}$. This motivates the approach of training a neural network $\boldsymbol{E}_{\tau_i}^{\tau^*} : \mathbb{R}^{\tau^* d} \to \mathbb{R}^d$ to approximate the mapping $\boldsymbol{\mu}_{\tau_{i+1}} = \boldsymbol{E}_{\tau_i}^{\tau^*}\left(\boldsymbol{\mu}_{\tau_{i-(\tau^*-1)}}, \ldots, \boldsymbol{\mu}_{\tau_i}\right)$ which is an extrapolation procedure for computing $\boldsymbol{\mu}_{\tau_{i+1}}$. As the extrapolation function modulates over time, it is necessary to train over various samples of the stochastic weight values rather than considering weight values far back in time. Thus, our training data $T_i^{\tau^*}$ is given by

$$T_i^{\tau^*} = \left\{ \left(\boldsymbol{w}_{\tau_{i-\tau^*}}(\omega_s), \ldots, \boldsymbol{w}_{\tau_{i-1}}(\omega_s) \right) \mapsto \boldsymbol{w}_{\tau_i}(\omega_s) \quad | s = 1, \ldots, N_s \right\} \tag{3.2}$$

where τ^* provides us with the time dependence and N_s the number of samples of the random variables obtained by repeatedly training sequences of DNNs on randomized subsamples of the microscale data. Note that training on $T_i^{\tau^*}$ approximates $\boldsymbol{E}_{\tau_{i-1}}^{\tau^*}$ with the assumption $\boldsymbol{E}_{\tau_i}^{\tau^*} \approx \boldsymbol{E}_{\tau_{i-1}}^{\tau^*}$ if the modulation of $\boldsymbol{\mu}_\tau$ is sufficiently slow. Letting $\widehat{\boldsymbol{\mu}}_{\tau_{i-(\tau^*-1)}} = \sum_{s=1}^{N_s} \boldsymbol{w}_{\tau_{i-(\tau^*-1)}}(\omega_s), \ldots, \widehat{\boldsymbol{\mu}}_{\tau_i} = \sum_{s=1}^{N_s} \boldsymbol{w}_{\tau_i}(\omega_s)$ be the sample means of the data, we extrapolate via

$$\boldsymbol{\mu}_{\tau_{i+1}} = \boldsymbol{E}_{\tau_{i-1}}^{\tau^*}\left(\widehat{\boldsymbol{\mu}}_{\tau_{i-(\tau^*-1)}}, \ldots, \widehat{\boldsymbol{\mu}}_{\tau_i}\right)$$

to obtain the next value of the mean at the next time step.

The behavior of neural networks on training data generated from some joint distribution is understood in the case of infinite data and can be used to estimate the function training will learn in the finite data case. Consider the extrapolation of a d-dimensional weight stochastic process $\boldsymbol{w}_\tau(\omega) \in \mathbb{R}^d$. The random vectors at a finite sequence of mesosteps $\tau_1, \ldots, \tau_n, \tau_{n+1}$ given by $\boldsymbol{w}_{\tau_1}(\omega), \ldots, \boldsymbol{w}_{\tau_n}(\omega), \boldsymbol{w}_{\tau_{n+1}}(\omega)$ are governed by some joint distribution such that training data $\{\boldsymbol{X}^q, \boldsymbol{t}^q\}_{q=1}^{N_s}$ given by N_s samples is distributed according to

$$\left\{ \left(\begin{bmatrix} \boldsymbol{w}_{\tau_1}(\omega_s) & \cdots & \boldsymbol{w}_{\tau_n}(\omega_s) \end{bmatrix}^T, \boldsymbol{w}_{\tau_{n+1}}(\omega_s) \right) \right\}_{s=1}^{N_s} \sim P^{(n+1)d}(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n, \boldsymbol{w}_{n+1})$$

where the data is generated by some underlying joint distribution $p(\boldsymbol{X}, \boldsymbol{t})$ with $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{w}_1 & \cdots & \boldsymbol{w}_n \end{bmatrix}^T \in \mathbb{R}^{nd}$ and $\boldsymbol{t} = \boldsymbol{w}_{n+1} \in \mathbb{R}^d$. We would like to find the parameters $\boldsymbol{\mathcal{W}}$ of a DNN function $\boldsymbol{f}(\boldsymbol{X}, \boldsymbol{\mathcal{W}}) : \mathbb{R}^{nd} \times \mathbb{R}^{D_{\boldsymbol{\mathcal{W}}}} \to \mathbb{R}^d$ that minimizes the sum of the squared error over training data. This is a cost function of the form. $E^s(\boldsymbol{\mathcal{W}}) = \frac{1}{2} \sum_{q=1}^{N_s} \sum_{k=1}^{d} [f_k(\boldsymbol{X}^q, \boldsymbol{\mathcal{W}}) - t_k^q]^2$. In the limit of infinite data, this cost function can be expressed as

$$E^s(\boldsymbol{w}) \;\; = \;\; \frac{1}{2} \sum_{k=1}^{d} \int \int [f_k(\boldsymbol{X}, \boldsymbol{\mathcal{W}}) - t_k^q]^2 p(\boldsymbol{X}, \boldsymbol{t}) \mathrm{d}\boldsymbol{X} \mathrm{d}\boldsymbol{t}$$

where the global minimum can be shown to be the function $\boldsymbol{f}$ with components

$$f_k(\boldsymbol{X}, \boldsymbol{\mathcal{W}}^*) = \langle t_k | \boldsymbol{X} \rangle_{p(\boldsymbol{t}|\boldsymbol{X})} = \langle p(\boldsymbol{t}|\boldsymbol{X}) \rangle_k$$

i.e., the network function components are given by components of the mean of the conditional distribution of the target data $\boldsymbol{t}$ conditioned on the input vector $\boldsymbol{X}$. When $P^{(n+1)d}(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n, \boldsymbol{w}_{n+1})$ takes the form of a normal distribution $\mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_1 & \cdots & \boldsymbol{\mu}_{n+1} \end{bmatrix}^T, \boldsymbol{\Sigma} \right)$, where $\boldsymbol{\Sigma} \in \mathbb{R}^{(n+1)d \times (n+1)d}$, the solution takes the form of a linear function

$$\boldsymbol{f}(\boldsymbol{W}_1 = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n]^T, \boldsymbol{\mathcal{W}}^*) = \langle p(\boldsymbol{w}_{n+1}) | \boldsymbol{w}_1, \ldots, \boldsymbol{w}_n \rangle = \boldsymbol{\mu}_2 + \boldsymbol{\mathcal{S}}_{21} \boldsymbol{\mathcal{S}}_{11}^{-1} (\boldsymbol{W}_1 - \boldsymbol{\mu}_1)$$

with $\boldsymbol{W} = [\boldsymbol{w}_1 \cdots \boldsymbol{w}_n \boldsymbol{w}_{n+1}]^T$ partitioned into $\boldsymbol{W}_1 = [\boldsymbol{w}_1 \cdots \boldsymbol{w}_n]^T$, $\boldsymbol{W}_2 = [\boldsymbol{w}_{n+1}]^T$ and μ, Σ partitioned accordingly into $\boldsymbol{\mu}_i, \boldsymbol{\mathcal{S}}_{ij}; i, j = 1, 2$. Observe that $\boldsymbol{f}(\boldsymbol{W}_1 = \boldsymbol{\mu}_1, \boldsymbol{\mathcal{W}}^*) = \boldsymbol{\mu}_2$ so that the image

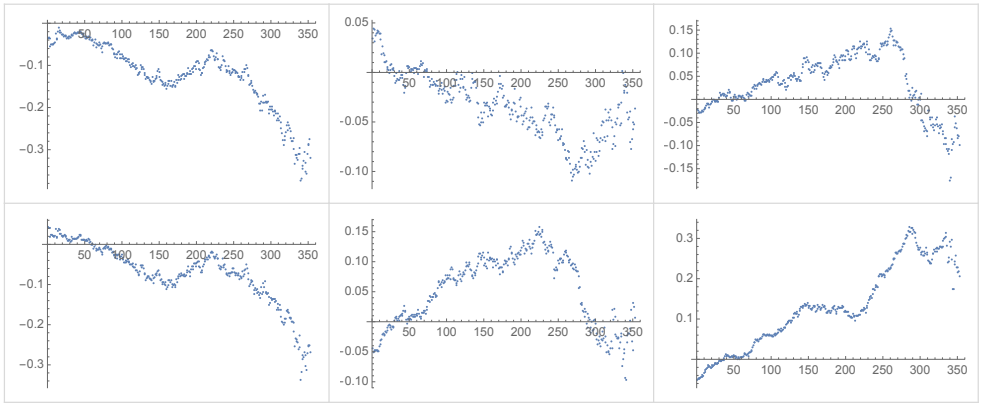
of the mean of the input is the mean of the output. An important special case occurs when $\boldsymbol{w}_{\tau_1}(\omega) \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \ldots, \boldsymbol{w}_{\tau_{n+1}}(\omega) \sim \mathcal{N}(\boldsymbol{\mu}_{n+1}, \boldsymbol{\Sigma}_{n+1})$ are independent so that the covariance matrix of the joint distribution is a diagonal block matrix $\boldsymbol{\Sigma} = \mathrm{diag}(\boldsymbol{\Sigma}_{11}, \ldots, \boldsymbol{\Sigma}_{(n+1)(n+1)})$. In this case, $\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}_2$ and $\bar{\boldsymbol{\Sigma}} = \boldsymbol{\mathcal{S}}_{22}$, i.e., we cannot improve our prediction knowing previous values.

3.3: RESULTS AND CONCLUSION

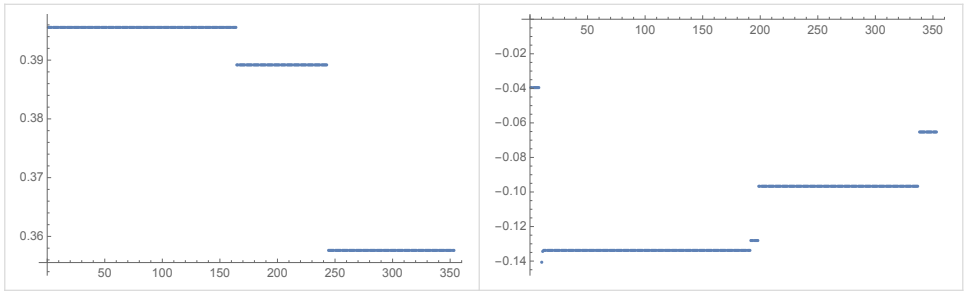
A sequence of mesoscale forward maps $\boldsymbol{F}_M(\mathbf{x}, \boldsymbol{\mathcal{W}}_M)$ were trained at successive mesosteps $\tau_M = M\Delta\tau$, $M = 1, \ldots, N_{\text{meso}}$ where the mesoscale is defined as in the previous chapter. The weights from each linear layer of the forward map DNNs were extracted as time series with the weight values modulating at the mesoscale.

3.3.1 Examination of the weights

Examination of the weights reveals that many exhibit the structure of a continuous stochastic process and are potentially amenable to Many of the weights in the intermediate layers of the network display constant or quasi-constant behavior switching between several fixed values. Examples of this behavior are shown in figure 3.4.



(a) Continuous modulation



(b) Quasi-constant modulation

Figure 3.4: Visualizing the behavior of temporally varying weights.

The weights in the encoding and decoding layers exhibit the greatest variance with the intermediate layer weights displaying relatively little modulation that increases toward the final layers of the network. This behavior is depicted in figure 3.5 where the singular values of the weight data from each linear layer are displayed.

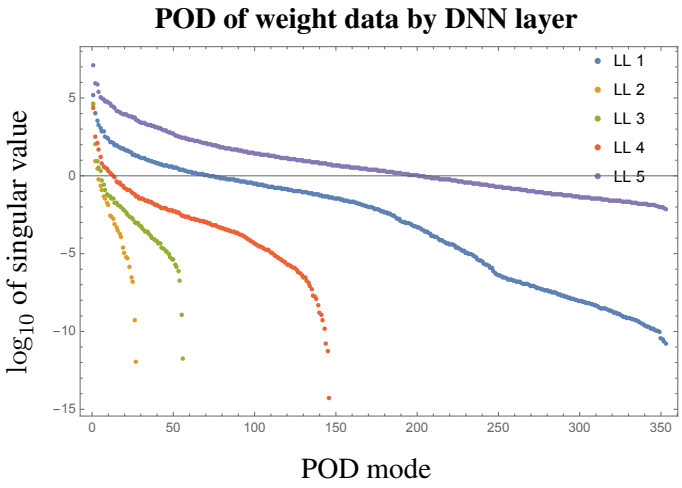


Figure 3.5: Typical behavior of weight sequences from the mesoscale forward map DNN

The decay in singular values is slowest in the final decoding layer followed by the encoding layer. This indicates that the encoding and decoding components are the most important components for capturing the slow modulation of the system. The greater variance in the decoding layer is accounted for by the fact that given a projection into a lower dimensional space, there are many corresponding lifting operations mapping back into the high-dimensional space.

3.3.2 Stochastic process extrapolation

Gaussian process extrapolation was applied to a sequence of DNNs approximating the mesoscale forward map of the MD system at a series of times. To gauge the effectiveness of the extrapolation procedure, the rate of convergence to a local minimum during SGD was compared to the cases of using constant extrapolation and random sampling from a distribution to initialize the weights. On average, Gaussian process extrapolation performs equivalently to constant extrapolation and both methods are significantly faster than random initialization of the weights.

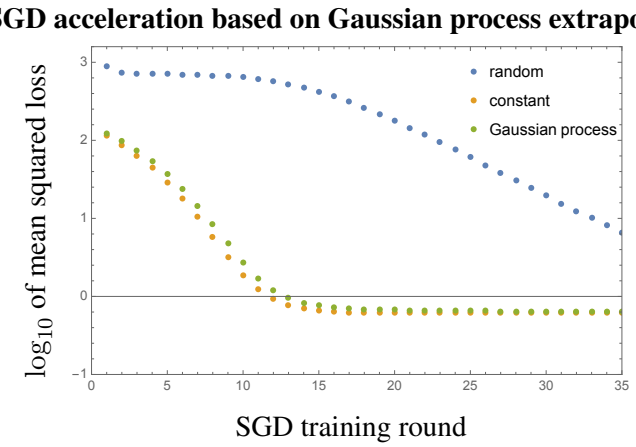


Figure 3.6: SGD convergence of Gaussian process extrapolation.

No significant acceleration of SGD convergence is provided by Gaussian extrapolation suggesting the weights are not Gaussian stochastic processes and a non-Gaussian approach may offer a better prediction. The least squares procedure extrapolates only the first moment and offers a basic non-Gaussian procedure. This method was again compared to constant and random extrapolation on the same sequence of DNNs to determine the acceleration potential. On average, the least squares approach performed equally well to constant extrapolation but provided no significant acceleration of SGD convergence.

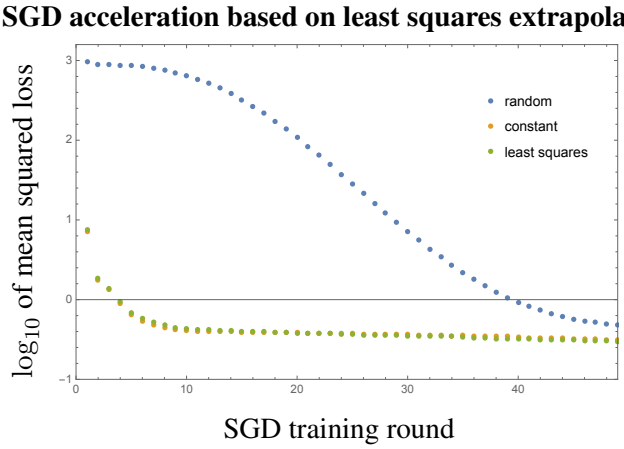


Figure 3.7: SGD convergence of least squares extrapolation of the first moment.

In addition to least-squares extrapolation, DNNs were also considered as a method to predict future values of the first moment of the weight stochastic processes. The DNN extrapolation accelerated SGD

compared to random initialization but was provided a significantly worse prediction of a local minimum than using constant extrapolation.

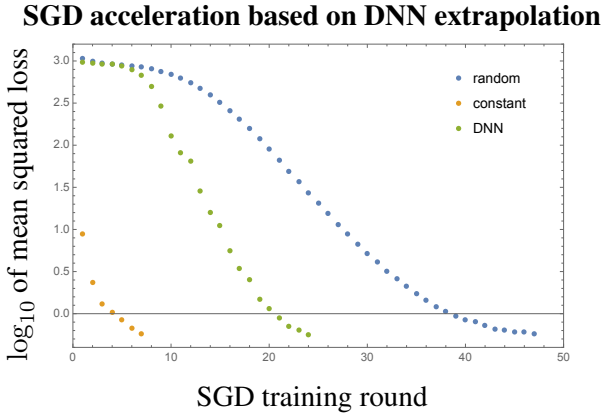


Figure 3.8: SGD convergence of least squares extrapolation of the first moment.

The training data for the DNNs is given by equation (3.2) and consists of multiple random samples of a given weight obtained by repeatedly carrying out SGD with the same training data. The poor performance of the DNN extrapolation function may result from the existence of many equivalent local minima of the cost function which are sampled by each randomly initialized SGD procedure and no effective functional relationship is learned by the network.

CHAPTER 4: NONLINEAR MODEL REDUCTION OF STOCHASTIC PROCESSES

4.1: INTRODUCTION

In chapter 3, a stochastic process representation of an SDE was extracted through a procedure based on DNNs. A natural question becomes whether a reduced description for the stochastic processes themselves can be obtained in some manner. In this chapter, a framework is developed for carrying out model reduction on stochastic processes by finding a reduced description of their distributions. Recall that a 1-dimensional stochastic process $X_t(\omega)$ is defined by its finite dimensional joint probability densities $p(x_n, t_n; \ldots; x_1, t_1)$ such that if $E = \{\omega \in \Omega | X_{t_j} \in A_j \text{for} 1 \leq j \leq n\}$ is an event, the probability that E occurs is given by $\int_A p(x_n, t_n; \ldots; x_1, t_1)\, \mathrm{d}x_1 \cdots \mathrm{d}x_n$ where $A = A_1 \times \cdots \times A_n$. Sampling values of X_t at a large number of times $t_1, \ldots, t_n$ is carried out by sampling the corresponding high-dimensional joint probability density. The computational complexity of sampling high-dimensional densities can be reduced by finding a lower-dimensional distribution whose samples can be transformed into approximate samples of the high-dimensional distribution.

4.2: MODEL REDUCTION OF GAUSSIAN PROCESSES IN EUCLIDEAN SPACE

This procedure can be described in the context of performing model reduction on a multivariate normal distribution governing a Gaussian stochastic process $X_t(\omega)$. Suppose that X_t is stationary with finite dimensional density

$$p(x_n, t_n; \ldots; x_1, t_1) = (2\pi)^{-\frac{n}{2}} \det(\boldsymbol{\Sigma})^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})} = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where $\mathbf{x} = [X_{t_1} \cdots X_{t_n}]^T$ and $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ are the mean and covariance, respectively. As $\boldsymbol{\Sigma}$ is symmetric, positive-definite, admits a unitary eigendecomposition $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ such that $\boldsymbol{\Sigma}^{-1} = \mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{U}^T$. Making the

coordinate transformation $\mathbf{y} = \mathbf{U}^T(\mathbf{x} - \boldsymbol{\mu})$ we have that

$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}) = \mathcal{N}(\mathbf{0}, \text{diag}(\sigma_1, \ldots, \sigma_n))$$

where $\sigma_1, \ldots, \sigma_r$ are much larger than the remaining eigenvalues $\sigma_{r+1}, \ldots, \sigma_n$. As $\sigma_{r+1}, \ldots, \sigma_n$ are small, we can reduce the dimensionality of the random vector $\mathbf{x}$ by projecting onto the subspace spanned by the first r eigenvectors $[\mathbf{u}_1 \cdots \mathbf{u}_r]^T = \mathbf{U}_r$ of the covariance matrix. The projection of $\mathbf{x}$ onto the affine subspace $\boldsymbol{\mu} + \text{span}\{\mathbf{u}_1 \cdots \mathbf{u}_r\}$ is given by

$$\boldsymbol{\mu} + \mathbf{U}_r\mathbf{U}_r^T(\mathbf{x} - \boldsymbol{\mu}) = \boldsymbol{\mu} + \mathbf{U}\mathbf{P}_r\mathbf{P}_r^T\mathbf{U}(\mathbf{x} - \boldsymbol{\mu}) = \boldsymbol{\mu} + \mathbf{U}\mathbf{P}_r\mathbf{P}_r^T\mathbf{y}$$

where $\mathbf{P}_r = [\mathbf{e}_1 \cdots \mathbf{e}_r]$ so that it suffices to project $\mathbf{y}$ onto the first r standard basis vectors and then apply the appropriate rotation and translation. Observe that the projection of $\mathbf{y}$ is distributed like

$$\mathbf{P}_r\mathbf{P}_r^T\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_r\text{diag}(\sigma_1, \ldots, \sigma_r)\mathbf{P}_r^T) = \mathcal{N}(\mathbf{0}, \text{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)) \tag{4.1}$$

which is a degenerate distribution of the form

$$\frac{1}{(2\pi)^{r/2}\sqrt{\prod_{i=1}^r \sigma_i}} \prod_{j=1}^{r} \exp\left(-\frac{y_j^2}{2\sigma_j}\right) \prod_{k=r+1}^{n} \delta(y_k) \tag{4.2}$$

representing a random vector $\mathbf{y}$ whose first r components are independently Gaussian distributed and whose remaining components $y_{r+1}, \ldots, y_n$ are effectively deterministic taking on their mean value of zero with probability one. Model reduction is achieved by approximating samples of $\mathbf{x}$ through samples of $\boldsymbol{\mu} + \mathbf{U}\mathbf{P}_r\mathbf{P}_r^T\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$.

To see that projection onto lower-dimensional, degenerate distribution can be used to approximate a high-dimensional distribution, consider a random variable $X(\omega) \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, $X(\omega), \boldsymbol{\mu} \in \mathbb{R}^{100}$ where $\Sigma = \mathbf{U}\boldsymbol{\Lambda}^T\mathbf{U}$ such that the eigenvalues of $\boldsymbol{\Sigma}$ are ordered by magnitude along the diagonal of $\boldsymbol{\Lambda}$ and represent the variances along the principal axes of a hyperellipse defined by the columns of $\mathbf{U}$. Suppose that the variances $(\sigma_1, \ldots, \sigma_{100}) = \text{diag}(\boldsymbol{\Lambda})$ decay exponentially according to figure 4.1.

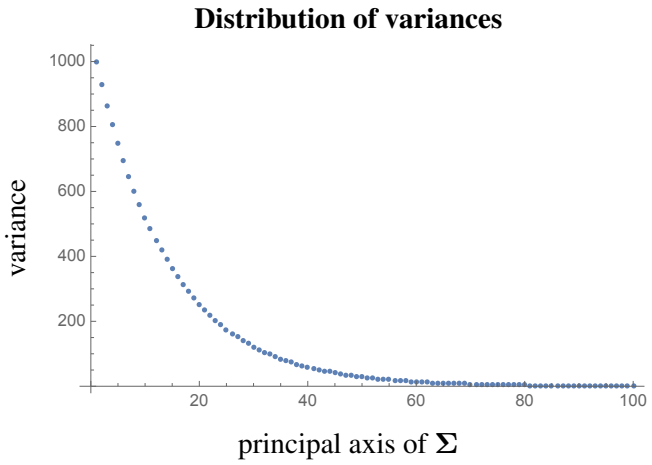


Figure 4.1: Distribution of the variances along the principal axes $\mathbf{U}$ of $\boldsymbol{\Sigma}$

We can then project onto a lower-dimensional Gaussian $\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ and extract the projection error as a function of r. The error measure is taken to be the relative error $\frac{\|\boldsymbol{\Sigma}-\hat{\boldsymbol{\Sigma}}_r\|}{\|\boldsymbol{\Sigma}\|}$ between the original covariance $\boldsymbol{\Sigma}$ and the estimated covariance $\hat{\boldsymbol{\Sigma}}_r$ from data sampled according to the degenerate distribution $\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$. The relative error as a function of r is depicted in figure 4.2.

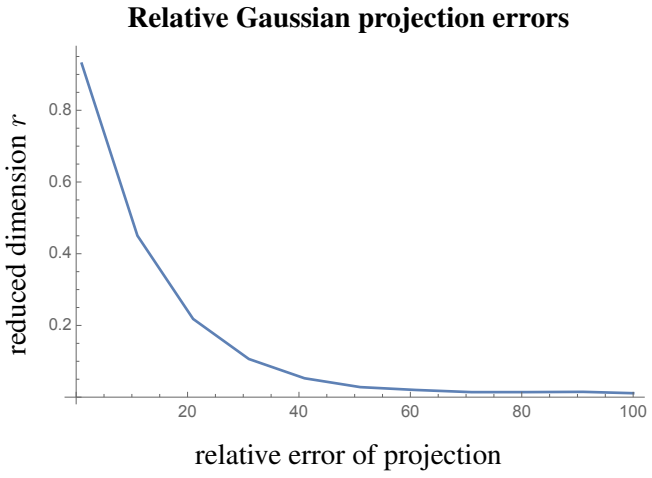


Figure 4.2: Distribution of the variances along the principal axes $\mathbf{U}$ of $\boldsymbol{\Sigma}$

Observe that the relative error becomes small when the dominant variances, given approximately by the 60 largest values, are accounted for in the projected distribution. This procedure is essentially carrying out PCA directly on the multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Note that the reduction in dimension was obtained through linear projection onto a subspace with the standard projection operator $\mathbf{P}_r\mathbf{P}_r^T$ which implicitly considers the coordinates $\sigma_1, \ldots, \sigma_n$ as describing a point in Euclidean space. Projection then minimizes the L^2 norm between the original point and its projection which does not reflect a natural distance between probability distributions. Instead, it is more appropriate to consider families of distributions as forming points of a manifold with an underlying metric determined through information theoretic quantities that define a more statistically informative notion of distance. This approach is known as information geometry and is discussed in the subsequent sections.

4.3: MODEL REDUCTION ON STATISTICAL MANIFOLDS

Information geometry [6] re-frames statistical problems in terms of differential geometry.An overview of relevant elementary differential geometry is provided in Appendix A. This approach allows for the transfer geometric notions, such as projection onto a lower dimensional subspace, to families of probability distributions. To obtain such a geometric space, a notion of distance is introduced to parametric families of probability distributions $S = \{p_\xi = p(x;\xi) | \xi = [\xi^1, \ldots, \xi^n \in \Xi]\}$ of the form

$$g_{ij}(\xi) = \mathbb{E}_\xi[\partial_i \log p(\boldsymbol{x};\xi)\partial_j \log p(\boldsymbol{x};\xi)]$$

This distance, known as Fisher information metric [6, 4, 5, 3] is based on information theory and defines an n-dimensional Riemannian manifold with a metric. This structure allows the computation of Fisher distance on the manifold and induces a Levi-Cevita connection $(\Gamma_{ij}^k)_p$ producing geodesic curves

$$\ddot{\gamma}^k(t) + (\Gamma_{ij}^k)_{\gamma(t)}\dot{\gamma}^i(t)\dot{\gamma}^j(t) = 0$$

that minimize the Fisher distance between points. Statistical problems can then be framed and solved in a geometric manner. An important operation from the point of view of information geometry which generalizes from vector spaces to curved manifolds is projection onto a subspace. In the vector space context, projection of a point finds a corresponding point in a lower-dimensional, linear subspace of minimal distance. Taking the distance as the L^2 norm, this can be seen as geodesic transport along a straight line intersecting the subspace orthogonally. This operation generalizes to projection onto a curved submanifold and in special cases this looks like geodesic transport along a curve whose intersection with the submanifold locally has the same

orthogonal structure. In the context of information geometry, projection can be used as a model reduction technique to approximate a high-dimensional, multivariate probability distribution as a simpler distribution of minimal distance with respect to information content. This has many potential applications such as dimension reduction of stochastic processes through information geometric projection of their finite dimensional joint distributions.

4.4: INFORMATION THEORY

Information theory plays an important role in the geometry of statistical manifolds. It provides a notion of distance between probability distributions that reflects the change in information content and is more statistically informative than a standard L^2 norm between functions.

Information theory can be described in the context of a discrete random variable $X : \Omega \to \mathbb{M}$ where $\mathbb{M} = \{x_1, \ldots, x_n\}$ is thought of as a finite space of messages that X can produce. The self-information of X at $x \in \mathbb{M}$ is given by $I_p(x) = -\log p(x)$ where $p(x)$ is the probability distribution of X. Intuitively, this represents the surprisal of seeing the message $x \in M$ and taking the expectation provides the entropy functional

$$\mathcal{H}(X) = \mathbb{E}_p[I_p(x)] = -\sum_{x \in \mathbb{M}} p(x) \log p(x)$$

measuring the uncertainty in the next message knowing only the distribution $p(x)$. It is maximized by a uniform distribution $p(x) = \frac{1}{n}$ and minimized by a distribution $p(x_i) = 1, p(x_j) = 0; j \neq i$ with mass concentrated at a single point. This can be extended to continuous random variables X with probability density functions $p(x; \xi)$ parametrized by $\xi \in \mathbb{R}$ to obtain an entropy functional of the form

$$\mathcal{H}(X) = \mathbb{E}_p[I_p(x)] = -\int p(x; \xi) \log p(x; \xi) \mathrm{d}x$$

maximized and minimized analogously by a uniform and delta distributions, respectively. Two continuous random variables X, Y goverened by the joint distribution $p(x, y; \xi)$ have joint entropy

$$\mathcal{H}(X, Y) = \mathbb{E}_p[I_p(x, y)] = -\int p(x, y; \xi) \log p(x, y; \xi) \mathrm{d}x \mathrm{d}y$$

Taking the expectation of the difference in self-formation between two distributions $p(x), q(x)$, respectively, provides the Kullback-Liebler divergence between the distributions

$$\mathcal{D}(p \prec q) = \mathbb{E}_p[I_p - I_q]$$

This provides a useful measure of the difference between two distributions that is not symmetric $\mathcal{D}(p \prec q) \neq \mathcal{D}(q \prec p)$ and so does not provide a notion of distance in the precise sense. An initial approach for constructing a metric on a parametric family of distributions $p(x;\xi)$ could be to consider the expected change in information $\mathbb{E}_p\left[\frac{\partial I_p}{\partial \xi}\right]$ as function of the parameter ξ. Observe that

$$\mathbb{E}_p\left[\frac{\partial I_p}{\partial \xi}\right] = \int p(x;\xi)\frac{\partial}{\partial \xi}\log p(x;\xi)\mathrm{d}x = \frac{\partial}{\partial \xi}\int p(x;\xi)\mathrm{d}x = 0$$

so that this definition is not useful. Instead, the Fisher information $F(\xi)$ is defined as the variance of the change in information

$$F(\xi) = \mathbb{E}_p\left[\left(\frac{\partial I_p}{\partial \xi}\right)^2\right] = -\mathbb{E}_p\left[\frac{\partial^2}{\partial^2 \xi}\log p(x;\xi)\right]$$

which can be thought of as the average curvature over x of $I_p(x)$ with respect to ξ.

4.5: STATISTICAL MANIFOLDS

A n-dimensional statistical manifold [6] built on top of a family of probability distributions $p(x;\boldsymbol{\xi})$ called a statistical model S defined as the set

$$S = \{p(x;\boldsymbol{\xi})|\boldsymbol{\xi} = [\xi^1, \ldots, \xi^n] \in \Xi\}$$

with $\Xi \subseteq \mathbb{R}^n$ and the mapping $\boldsymbol{\xi} \mapsto p(x;\boldsymbol{\xi})$ injective. By considering parameterizations which are $\mathcal{C}^\infty$ compatible to be equivalent, we can consider S to be a smooth manifold. As an example, consider the manifold $S(\mathcal{N}_k)$ of k-dimensional multivariate normal distributions $\mathcal{N}_k$ defined by

$$n = k + \frac{k(k+1)}{2}, \boldsymbol{\xi} = [\boldsymbol{\mu}, \boldsymbol{\Sigma}], \Xi = \left\{[\boldsymbol{\mu}, \boldsymbol{\Sigma}]|\boldsymbol{\mu} \in \mathbb{R}^k, \boldsymbol{\Sigma} \in \mathbb{R}^{k\times k}\text{positive definite}\right\}$$

$$p(\mathbf{x};\boldsymbol{\xi}) = \frac{(2\pi)^{-k/2}}{(\det \boldsymbol{\Sigma})^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

Given multidimensional parameters $\boldsymbol{\xi} = [\xi^1, \ldots, \xi^n]$, the Fisher information takes the form of a matrix

$$g_{ij}(\boldsymbol{\xi}) = \mathbb{E}_{p(x,\boldsymbol{\xi})}[\partial_i \log p(x;\boldsymbol{\xi}) \partial_j \log p(x;\boldsymbol{\xi})] \tag{4.3}$$

which defines a scalar product $\left\langle a^i \left(\frac{\partial}{\partial \xi^i}\right)_p, b^i \left(\frac{\partial}{\partial \xi^i}\right)_p \right\rangle_p = g_{ij}(\boldsymbol{\xi}(p)) a^i b^j$ on each tangent space T_pM. This scalar product induces the norm $\|\cdot\|_p : T_pM \to \mathbb{R}_{\geqslant 0}$ on T_pM defined by $\sqrt{\langle U, U \rangle_p}$. A distance between two points $p, q \in M$ can then be defined as

$$\delta(p,q) = \inf \left\{ \int_0^1 \|\dot{\gamma}(t)\|_{\gamma(t)} \mathrm{d}t : \gamma \in \mathcal{C}^1([0,1]), \gamma(0) = p, \gamma(1) = q \right\}$$

where $\dot{\gamma}(t) = \dot{\gamma}^i(t) \left(\frac{\partial}{\partial \xi^i}\right)_{\gamma(t)}$ is the velocity of the curve in local coordinates.

4.6: MODEL REDUCTION OF GAUSSIAN PROCESSES ON STATISTICAL MANIFOLDS

In this section, we begin developing techniques for model reduction of multivariate Gaussian distributions through geodesic projection onto submanifolds of a statistical manifold. The Fisher metric (4.3) defines geodesics which minimize Fisher information distance and can be used to formulate a notion of projection on a manifold.

4.6.1 Projection on the univariate Gaussian manifold

It is instructive to consider compare how projection on the manifold of univariate Gaussian distributions $S(\mathcal{N}_1) = \{\mathcal{N}(\mu, \sigma) | \mu \in \mathbb{R}, \sigma \in \mathbb{R}_{>0}\}$ distributions differs from standard orthogonal projection. Finding the Fisher distance on N_1 can be done by first computing the Fisher information metric which has the form

$$g_{ij}(\mu, \sigma) = \begin{bmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{2}{\sigma^2} \end{bmatrix} \tag{4.4}$$

This can be seen to be similar to the Poincaré half-plane and has geodesics given by vertical straight lines and half-ellipses centered at $\sigma = 0$. Minimal geodesic distance can be computed with the closed form

expression

$$d_{N_1}((\mu_1,\sigma_1),(\mu_2,\sigma_2)) = \sqrt{2}\ln\frac{\left\|(\frac{\mu_1}{\sqrt{2}},\sigma_1)-(\frac{\mu_2}{\sqrt{2}},-\sigma_2)\right\|+\left\|(\frac{\mu_1}{\sqrt{2}},\sigma_1)-(\frac{\mu_2}{\sqrt{2}},\sigma_2)\right\|}{\left\|(\frac{\mu_1}{\sqrt{2}},\sigma_1)-(\frac{\mu_2}{\sqrt{2}},-\sigma_2)\right\|-\left\|(\frac{\mu_1}{\sqrt{2}},\sigma_1)-(\frac{\mu_2}{\sqrt{2}},\sigma_2)\right\|} \tag{4.5}$$

Projection onto a lower-dimensional submanifold can be carried out through the appropriate minimization of d_{N_1}. As an example, consider the problem of estimating the distribution of an underlying Gaussian random variable $X(\omega) \sim \mathcal{N}(\mu', \sigma')$. Suppose that an estimate $\mathcal{N}(\hat{\mu}, \hat{\sigma})$ of the distribution has been obtained through data and the true mean μ' is known based on underlying physical constraints of the problem. Correction of the estimate $\mathcal{N}(\hat{\mu}, \hat{\sigma})$ can be expressed as finding new parameters (μ', σ') such that $d_{N_1}((\hat{\mu}, \hat{\sigma}), (\mu', \sigma'))$ is minimized. This is simply projection onto the 1-dimensional submanifold of N_1 defined by constraining the mean coordinate at $\mu = \mu'$ and is depicted in figure 4.3.

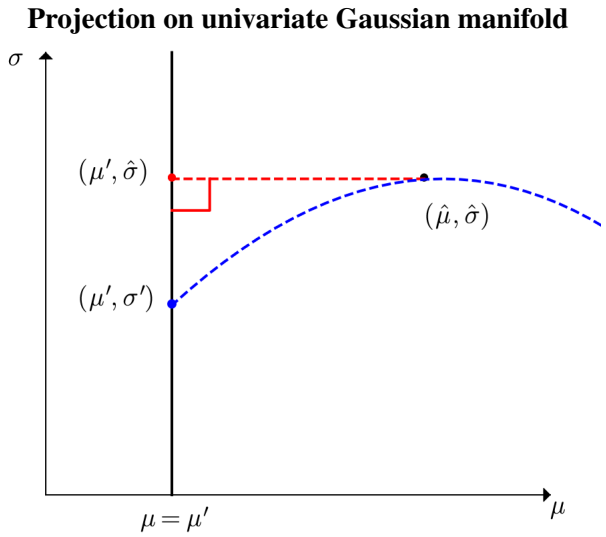


Figure 4.3: Orthogonal (red) projection versus actual minimal geodesic projection (blue).

Observe that projection in the curved space of univariate Gaussian distributions differs from standard Euclidean projection but still provides element of minimal distance from a given point that lies in a given subspace.

4.6.2 Projection on higher-dimensional Gaussian manifolds

For a multivariate Gaussian distribution in the general n-dimensional case parameterized by $\boldsymbol{\xi} = [\xi_1 \cdots \xi_n] \in \mathbb{R}^n$ such that $p(\mathbf{x};\boldsymbol{\xi}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\xi}), \boldsymbol{\Sigma}(\boldsymbol{\xi}))$ the Fisher metric has the following closed form expression [6]

$$g_{ij}(\boldsymbol{\xi}) = \frac{\partial \boldsymbol{\mu}^T}{\partial \xi_i} \boldsymbol{\Sigma}^{-1} \frac{\partial \boldsymbol{\mu}}{\partial \xi_j} + \frac{1}{2} \text{tr} \left(\boldsymbol{\Sigma}^{-1} \frac{\partial \boldsymbol{\Sigma}}{\partial \xi_i} \boldsymbol{\Sigma}^{-1} \frac{\partial \boldsymbol{\Sigma}}{\partial \xi_j} \right) \tag{4.6}$$

which induces a complex geometric structure on $S(\mathcal{N}_n)$ for even small n. Hence, it is necessary to consider at first submanifolds of $S(\mathcal{N}_n)$ where the moments $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ have special structure such that the metric defined by (4.6) simplifies considerably. Several such submanifolds are considered in the subsequent sections and a procedure for geodesic projection is developed for each.

4.6.3 Principal-component submanifolds of Gaussian statistical manifolds

Consider the finite-dimensional statistical manifold of m-variate normal probability density functions (PDFs)

$$S(\mathcal{N}_m) = \{\mathcal{N}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma}) | \boldsymbol{\mu} \in \mathbb{R}^m, \boldsymbol{\Sigma} \in \mathbb{R}^{m \times m} s.p.d.\}, \tag{4.7}$$

$$\mathcal{N}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})(\boldsymbol{x}) = \frac{1}{(2\pi)^{m/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left[-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right], \tag{4.8}$$

and denote by $\boldsymbol{\theta}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \in \mathbb{R}^n$, $n = m + m(m+1)/2$, the natural parametrization on this manifold formed from the mean vector $\boldsymbol{\mu}$, and the lower triangle of $\boldsymbol{\Sigma}$

$$\theta_l = \mu_l \quad , \quad l = 1, \ldots, m; \tag{4.9}$$

$$\theta_l = \Sigma_{jk} \quad , \quad j = 1, \ldots, m, b = 1, \ldots, j, m = k + (j-1)j/2 + k. \tag{4.10}$$

Introduce an associated information measure

$$i_m^{\mathcal{N}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})(\boldsymbol{x}) = -\log \mathcal{N}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})(\boldsymbol{x}) = \frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) + \frac{1}{2} \log |\boldsymbol{\Sigma}| + c_m, c_m = \frac{m}{2} \log(2\pi). \tag{4.11}$$

Reduction of m-variate data to principal components by unitary diagonalization of the covariance matrix $\boldsymbol{\Sigma} = \mathbf{U} \boldsymbol{\Lambda} \mathbf{U}^T$, induces the coordinate transformation $\boldsymbol{y} = \mathbf{U}^T (\boldsymbol{x} - \boldsymbol{\mu})$, in which the probability density and

information measure are conveniently expressed as

$$\mathcal{N}_m(\mathbf{0},\mathbf{\Lambda})(\boldsymbol{y}) = \frac{1}{(2\pi)^{m/2}\left(\prod_{j=1}^{m}\lambda_j\right)^{1/2}}\exp\left[-\frac{1}{2}\boldsymbol{y}^T\mathbf{\Lambda}^{-1}\boldsymbol{y}\right], \tag{4.12}$$

$$i_m^{\mathcal{N}}(\mathbf{0},\mathbf{\Lambda})(\boldsymbol{y}) = \frac{1}{2}\boldsymbol{y}^T\mathbf{\Lambda}^{-1}\boldsymbol{y} + \frac{1}{2}\sum_{j=1}^{m}\log(\lambda_j) + c_m, \mathbf{\Lambda} = \mathrm{diag}(\lambda_1,\ldots,\lambda_m), \lambda_j > 0. \tag{4.13}$$

The set of principal component PDFs form an m-dimensional submanifold $P(\mathcal{N}_m)$

$$P(\mathcal{N}_m) = \{\mathcal{N}_m(\mathbf{0},\mathbf{\Lambda})|\mathbf{\Lambda} = \mathrm{diag}(\lambda_1,\ldots,\lambda_m) \in \mathbb{R}_+^{m\times m}\}, \tag{4.14}$$

of the n-dimensional statistical manifold of general multivariate normal distributions.

4.6.3.1 Metric and geodesics

The metric is diagonal and the inverse is readily computed as $g^{jj} = 2\lambda_j^2$. The Christoffel coefficients are given by $\Gamma_{kl}^i = \frac{1}{2}g^{im}(g_{mk,l} + g_{ml,k} - g_{kl,m}) = -\frac{1}{\lambda_i}\delta_{ik}\delta_{kl}$ and yield the geodesic equations

$$\ddot{\lambda}_i - \frac{\dot{\lambda}_i^2}{\lambda_i} = 0 \tag{4.15}$$

with solution $\lambda_i(t) = c_2 e^{c_1 t}; i = 1,\ldots,m$.

4.6.3.2 Geodesic projection

Given the simple form of the geodesics (4.15), we can find a closed form for the distance between two points which can yield a solution to projection through minimization. The geodesic corresponding the boundary value problem

$$\begin{cases} \ddot{\lambda}_i - \frac{\dot{\lambda}_i^2}{\lambda_i} &= 0 \\ \lambda_i(0) &= \alpha_i \\ \lambda_i(1) &= \beta_i \end{cases}$$

is then defined by by $\lambda_i(t) = \alpha_i \left(\frac{\beta_i}{\alpha_i}\right)^t$. The length of this geodesic is given by

$$\int_0^1 \left\| \frac{\mathrm{d}\lambda}{\mathrm{d}t} \right\| \mathrm{d}t = \int_0^1 \sqrt{g_{ij}\dot{\lambda}_i \dot{\lambda}_j} \mathrm{d}t = \frac{1}{2} \sqrt{\sum_{i=1}^m \left(\log \frac{\beta_i}{\alpha_i}\right)^2} \tag{4.16}$$

Consider the problem of projection of a distribution in $P(\mathcal{N}_m)$ defined by m coordinates $\alpha = (\alpha_1, \ldots, \alpha_m)$ onto the k-dimensional submanifold defined by $\{\boldsymbol{c} \in \mathbb{R}^m | c_{k+1} = \cdots = c_m = \varepsilon \in \mathbb{R}_{>0}\}$ where ε is typically small. This can be expressed as finding the end point of a minimal length geodesic which starts at $\alpha = (\alpha_1, \ldots, \alpha_m)$ and ends in the submanifold at some point $(\bar{\beta}_1, \ldots, \bar{\beta}_k, \varepsilon, \ldots, \varepsilon)$. Thinking of α as fixed, we can consider the geodesic distance function defined by

$$d(\boldsymbol{\beta}) \equiv \frac{1}{2} \sqrt{\sum_{i=1}^m \left(\log \frac{\beta_i}{\alpha_i}\right)^2}$$

and compute the projection via minimization of $d(\boldsymbol{\beta})$ with the constraints that $\beta_{k+1} = \cdots = \beta_m = \varepsilon$. This minimization problem can be expressed in terms of Lagrange multipliers as

$$\begin{cases} \nabla d(\boldsymbol{\beta}) - \sum_{i=k+1}^n \gamma_i \nabla g_i(\boldsymbol{\beta}) = 0 \\ g_{k+1}(\boldsymbol{\beta}) = \cdots = g_n(\boldsymbol{\beta}) = 0 \end{cases}$$

where $g_i(\boldsymbol{\beta}) = \beta_i - \varepsilon$ for $i = k+1, \ldots, n$. A solution is provided by solving $\frac{\partial d}{\partial \beta_i} = 0$ for $i = 1, \ldots, k$ with $\beta_{k+1} = \cdots = \beta_n = \varepsilon$. The projection of $\alpha = (\alpha_1, \ldots, \alpha_m)$ is then simply $(\alpha_1, \ldots, \alpha_k, \varepsilon, \ldots, \varepsilon)$. Projection in $P(\mathcal{N}_m)$ behaves the same way as Euclidean projection suggesting that $P(\mathcal{N}_m)$ is not capturing the underlying curved geometry of $S(\mathcal{N}_m)$. Hence, we consider an extension of $P(\mathcal{N}_m)$ in the next section and attempt to understand geodesic projection in that case.

4.6.4 Extension of principal component submanifolds

In the previous section we considered the variable transformation $\boldsymbol{y} = \mathbf{U}^T(\boldsymbol{x} - \boldsymbol{\mu})$ which led to the principal component Gaussian distributions

$$\mathcal{N}_m(\mathbf{0}, \boldsymbol{\Lambda})(\boldsymbol{y}) = \frac{1}{(2\pi)^{m/2} \left(\prod_{j=1}^m \lambda_j\right)^{1/2}} \exp\left[-\frac{1}{2} \boldsymbol{y}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{y}\right],$$

and the associated submanifold $P(\mathcal{N}_m) = \{\mathcal{N}_m(\mathbf{0}, \mathbf{\Lambda}) | \mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_m) \in \mathbb{R}_+^{m \times m}\}$ of the Gaussian statistical manifold. The basic geometry of this submanifold was computed.

In this section, we wish to develop an extension of the previous geometry to submanifold which again considers diagonal covariance matrices but with nonzero means. Hence we begin with the finite-dimensional statistical manifold $S(\mathcal{N}_m)$of m-variate Gaussian distributions with dimension $n = m + \frac{m(m+1)}{2}$. Introduce the unitary diagonalization of the covariance matrix $\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ along with the coordinate transformations $\boldsymbol{z} = \mathbf{Q}^T\boldsymbol{x}$ and $\boldsymbol{\nu} = \mathbf{Q}^T\boldsymbol{\mu}$ which allows us to express the probability density in the form

$$\mathcal{N}_m(\boldsymbol{\nu}, \mathbf{\Lambda})(\boldsymbol{z}) = \frac{1}{(2\pi)^{m/2}\left(\prod_{j=1}^m \lambda_j\right)^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{\nu})^T\mathbf{\Lambda}^{-1}(\boldsymbol{z}-\boldsymbol{\nu})\right]$$

The associated information measure is then given by

$$i_m^{\mathcal{N}}(\boldsymbol{\nu}, \mathbf{\Lambda})(\boldsymbol{z}) = -\log \mathcal{N}_m(\boldsymbol{\nu}, \mathbf{\Lambda})(\boldsymbol{z}) = \frac{1}{2}(\boldsymbol{z}-\boldsymbol{\nu})^T\mathbf{\Lambda}^{-1}(\boldsymbol{z}-\boldsymbol{\nu}) + \frac{1}{2}\sum_{j=1}^m \log(\lambda_j) + c_m$$

The set of such probability distributions forms a $2m$-dimensional submanifold

$$P^*(\mathcal{N}_m) = \left\{\mathcal{N}_m(\boldsymbol{\nu}, \mathbf{\Lambda}) \,\middle|\, \boldsymbol{\nu} \in \mathbb{R}^m; \mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_m) \in \mathbb{R}_+^{m \times m}\right\}$$

of the n-dimensional manifold of Gaussian PDFs.

4.6.4.1 Metric and geodesics

From the preceding section we see that the metric tensor is diagonal

$$\boldsymbol{G} = \text{diag}\left(\frac{1}{\lambda_1}, \ldots, \frac{1}{\lambda_m}, \frac{1}{2\lambda_1^2}, \ldots, \frac{1}{2\lambda_m^2}\right) \tag{4.17}$$

and induce the following geodesic equations. For $1 \leqslant i \leqslant m$ we have

$$\ddot{\xi}^i - \frac{\dot{\xi}^i\dot{\xi}^{i+m}}{\xi^{i+m}} = 0 \tag{4.18}$$

and for $m+1 \leqslant i \leqslant 2m$

$$\ddot{\xi}^i + (\dot{\xi}^{i-m})^2 - \frac{(\dot{\xi}^i)^2}{\xi_i} = 0 \tag{4.19}$$

4.6.5 Geodesic projection

We consider projection onto submanifolds in the case of $P^*(\mathcal{N}_m)$ of the form $\xi_{2m} = \cdots = \xi_{2m-K+1} = \varepsilon$ where $(\nu_1, \ldots, \nu_m, \lambda_1, \ldots, \lambda_m) = (\xi_1, \ldots, \xi_{2m})$ and $m + 2 < 2m - K + 1 \leqslant 2m \Leftrightarrow 1 \leqslant K < m - 1$. This carries out model reduction of Gaussian PDFs by approximately setting the smallest variances to zero to construct a lower-dimensional degenerate distribution as in section 4.2. Hence, we set the K smallest variances to ε which approximates the corresponding components of the multivariate gaussian PDF becoming delta functions.

Let S be an n-dimensional manifold with coordinates $\varphi = [\xi^i] = (\xi^1, \ldots, \xi^n), \varphi : S \to \mathbb{R}^n$ and let $M \subset S$ be a k-dimensional submanifold of S defined as

$$M = \left\{ p \in S | \xi^i(p) = \varepsilon \in \mathbb{R}_+, k+1 \leqslant i \leqslant n \right\}$$

for some $1 \leqslant k < n$. Let $\gamma : \mathbb{R} \to S$ be a curve and define $\bar{\gamma} : \mathbb{R} \to \mathbb{R}^n$ by $\bar{\gamma} = \varphi \circ \gamma$ and $\gamma^i = \xi^i \circ \gamma$. Furthermore, let γ satisfy

$$\ddot{\gamma}^i(t) + (\Gamma^i_{kl})_{\gamma(t)} \dot{\gamma}^k(t) \dot{\gamma}^l(t) = 0, i = 1, \ldots, n$$

i.e. γ is a geodesic curve. We want γ to also satisfy the following three properties

i. $\bar{\gamma}(0) = \alpha \in \mathbb{R}^n$

ii. $\gamma^{k+1}(1) = \cdots = \gamma^n(1) = \varepsilon$

iii. $\dot{\gamma}(1) \in T_{\gamma(1)}(M)^\perp$

Hence, γ is the geodesic curve with initial value $p = \varphi^{-1}(\alpha)$ that intersects M orthogonally at $\gamma(1)$. Note that

$$T_{\gamma(1)}(M) = \left\{ c^i \left(\frac{\partial}{\partial \xi^i} \right)_{\gamma(1)} \middle| c^i \in \mathbb{R}, i = 1, \ldots, k \right\}$$

We can express condition (iii) as follows

$$\begin{aligned} \dot{\gamma}(1) \in T_{\gamma(1)}(M)^\perp &\Leftrightarrow \langle \dot{\gamma}(1), v \rangle_{g(\gamma(1))} = 0 \qquad \forall v \in T_{\gamma(1)}(M) \\ &\Leftrightarrow \left\langle \dot{\gamma}(1), \left(\frac{\partial}{\partial \xi^i} \right)_{\gamma(1)} \right\rangle_{g(\gamma(1))} = 0, i = 1, \ldots, k \end{aligned}$$

Let $n = 2m$, some $n \in \mathbb{N}$ and consider metric tensors g of the form

$$\operatorname{diag}\left(\frac{1}{\xi^{m+1}}, \ldots, \frac{1}{\xi^{2m}}, \frac{1}{2(\xi^{m+1})^2}, \ldots, \frac{1}{2(\xi^{2m})^2}\right)$$

so that in general

$$\langle \dot{\gamma}(1), v\rangle_{g(\gamma(1))} = g_{ij}(\gamma(1))\dot{\gamma}^i(1)v^j = \sum_{i=1}^{m} \frac{1}{\gamma^{i+m}(1)}\dot{\gamma}^i(1)v^i + \sum_{i=m+1}^{2m} \frac{1}{2(\gamma^i(1))^2}\dot{\gamma}^i(1)v^i$$

Hence we have

$$\left\langle \dot{\gamma}(1), \left(\frac{\partial}{\partial \xi^i}\right)_{\gamma(1)} \right\rangle_{g(\gamma(1))} = \begin{cases} \frac{\dot{\gamma}^i(1)}{\gamma^{i+m}(1)} & 1 \leqslant i \leqslant m \\ \frac{\dot{\gamma}^i(1)}{2(\gamma^i(1))^2} & m < i \leqslant 2m \end{cases}$$

Condition (iii) can then be written simply as

$$\dot{\gamma}^i(1) = 0, i = 1, \ldots, k$$

With constraints given by (i), (ii), and (iii) we have a total of $2n$ conditions for our ODE system which takes the form

$$\begin{cases} \ddot{\xi}^i\xi^{i+m} - \dot{\xi}^i\dot{\xi}^{i+1} = 0 & 1 \leq i \leq m \\ \ddot{\xi}^i\xi^i + (\dot{\xi}^{i-m})^2\xi^i - (\dot{\xi}^i)^2 = 0 & m+1 \leq i \leq 2m \\ \xi^i(0) = \alpha^i & 1 \leq i \leq 2m \\ \dot{\xi}^i(1) = 0 & 1 \leq i \leq s \\ \dot{\xi}^i(1) = \varepsilon & s < i \leq 2m \end{cases} \tag{4.20}$$

where $s \in \mathbb{N}$ with $m + 1 < s < 2m$ so that $\xi^{s+1}, \ldots, \xi^{2m}$ denote the insignificant variances which we will drive to some small $\varepsilon > 0$.

4.6.6 Shooting method solution to geodesic projection equations

We aim to solve the boundary value problem 4.20 which we first replace by the initial value problem

$$\begin{aligned} \ddot{\boldsymbol{\xi}}(t) &= \boldsymbol{F}(\boldsymbol{\xi}(t), \dot{\boldsymbol{\xi}}(t)), \\ \boldsymbol{\xi}(0) &= \boldsymbol{\alpha}, \\ \dot{\boldsymbol{\xi}}(0) &= \boldsymbol{u} \end{aligned} \tag{4.21}$$

where $\boldsymbol{F}$ is the vector function defining the ODE system in 4.20. Let $\boldsymbol{\xi}(t, \boldsymbol{u}) = (\xi^1(t, \boldsymbol{u}), \ldots, \xi^{2m}(t, \boldsymbol{u}))$ be the solution of (4.21) with initial velocities $\boldsymbol{u}$. The constraints on $\boldsymbol{u}$ are

$$\dot{\xi}^i(1; \boldsymbol{u}) = 0, \text{approximated as } \xi^i(1; \boldsymbol{u}) = \xi^i(1 - \Delta t; \boldsymbol{u}), \text{for } 1 \leqslant i \leqslant s$$

$$\xi^i(1; \boldsymbol{u}) = \varepsilon, \text{for } s + 1 \leqslant i \leqslant 2m$$

These constraints can be represented in matrix form as

$$G(\boldsymbol{\xi}_u(1 - \Delta t), \boldsymbol{\xi}_u(1)) = \mathbf{A}\boldsymbol{\xi}_u(1) + \mathbf{B}\boldsymbol{\xi}_u(1 - \Delta t) + \boldsymbol{c} = 0$$

where

$$\mathbf{A} = \text{diag}(1, \ldots, 1)$$

$$\mathbf{B} = \text{diag}(\underbrace{-1, \ldots, -1}_{s}, \underbrace{0, \ldots, 0}_{2m-s})$$

$$\boldsymbol{c} = (\underbrace{0, \ldots, 0}_{s}, \underbrace{-\varepsilon, \ldots, -\varepsilon}_{2m-s})$$

Hence, the problem becomes determination of $\boldsymbol{u} \in \mathbb{R}^{2m}$ such that $\Phi(\boldsymbol{u}) = 0$ where

$$\Phi(\boldsymbol{u}) = \mathbf{A}\boldsymbol{\xi}_u(1) + \mathbf{B}\boldsymbol{\xi}_u(1 - \Delta t) + \boldsymbol{c} \tag{4.22}$$

which we aim to solve via Newton iteration. This requires computation of both $\Phi(\boldsymbol{u})$ and $\Phi'(\boldsymbol{u})$ for a given $\boldsymbol{u}$. We can compute $\Phi(\boldsymbol{u})$ by invoking an ODE solver to approximate $\boldsymbol{\xi}_u(t)$ for $t \in [0, 1]$. To compute $\Phi'(\boldsymbol{u})$,

first consider

$$\Phi'(\boldsymbol{u}) = \frac{\partial \Phi}{\partial \boldsymbol{u}} = \mathbf{A}\frac{\partial \boldsymbol{\xi}(1,\boldsymbol{u})}{\partial \boldsymbol{u}} + \mathbf{B}\frac{\partial \boldsymbol{\xi}(1-\Delta t,\boldsymbol{u})}{\partial \boldsymbol{u}}$$

Thus, letting $\mathbf{Y}(t) = \frac{\partial \boldsymbol{\xi}(t,\boldsymbol{u})}{\partial \boldsymbol{u}} = \frac{\partial}{\partial \boldsymbol{u}}\boldsymbol{\xi}_u(t)$ we can see that $\mathbf{Y}(t)$ solves the following ODE

$$\ddot{\mathbf{Y}}(t) = \frac{\partial}{\partial \boldsymbol{u}}\ddot{\boldsymbol{\xi}}_u(t) = \frac{\partial}{\partial \boldsymbol{u}}\boldsymbol{F}(\boldsymbol{\xi}_u(t), \dot{\boldsymbol{\xi}}_u(t)) \tag{4.23}$$

where the matrix $\frac{\partial}{\partial \boldsymbol{u}}\boldsymbol{F}(\boldsymbol{\xi}_u(t), \dot{\boldsymbol{\xi}}_u(t))$ has entries

$$\begin{aligned}\left[\frac{\partial}{\partial \boldsymbol{u}}\boldsymbol{F}(\boldsymbol{\xi}_u(t), \dot{\boldsymbol{\xi}}_u(t))\right]_{i,j} &= \frac{\partial F_i(\boldsymbol{\xi}(t,\boldsymbol{u}), \dot{\boldsymbol{\xi}}(t,\boldsymbol{u}))}{\partial u_j} \\ &= \frac{\partial F_i}{\partial \xi^k}\frac{\partial \xi^k}{\partial u_j} + \frac{\partial F_i}{\partial \dot{\xi}^l}\frac{\partial \dot{\xi}^l}{\partial u_j}\end{aligned}$$

with initial conditions given by

$$\mathbf{Y}(0) = \frac{\partial \boldsymbol{\xi}(0,\boldsymbol{u})}{\partial \boldsymbol{u}} = \frac{\partial}{\partial \boldsymbol{u}}\boldsymbol{\alpha} = 0 \tag{4.24}$$

$$\dot{\mathbf{Y}}(0) = \frac{\partial}{\partial \boldsymbol{u}}\dot{\boldsymbol{\xi}}(0) = \frac{\partial}{\partial \boldsymbol{u}}\boldsymbol{u} = \mathbf{I} \tag{4.25}$$

We can then compute

$$\Phi'(\boldsymbol{u}) = \mathbf{A}\frac{\partial \boldsymbol{\xi}(1,\boldsymbol{u})}{\partial \boldsymbol{u}} + \mathbf{B}\frac{\partial \boldsymbol{\xi}(1-\Delta t,\boldsymbol{u})}{\partial \boldsymbol{u}} = \mathbf{A}\mathbf{Y}(1) + \mathbf{B}\mathbf{Y}(1-\Delta t) \tag{4.26}$$

The Newton iteration procedure for carrying out geodesic projection then becomes

1. Provide: initial guess $\boldsymbol{u}_0 = \boldsymbol{u}_n$, tolerance δ, number of newton iterations N, initial value $\boldsymbol{\alpha}$, significant values s, final ε.

2. Solve the initial value problem (4.21) with initial conditions $\boldsymbol{\xi}(0) = \boldsymbol{\alpha}$ and $\dot{\boldsymbol{\xi}}(0) = \boldsymbol{u}_n$.

3. Using the solution from step 2, solve the initial value problem (4.23) with initial conditions (4.24) and (4.25).

4. Compute $\Phi(\boldsymbol{u})$, $\Phi'(\boldsymbol{u})$ and then carry out a Newton iteration via

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \boldsymbol{d}_n$$

where $\boldsymbol{d}_n$ solves

$$\Phi'(\boldsymbol{u}_n)\boldsymbol{d}_n = -\Phi(\boldsymbol{u}_n)$$

5. Check if $\|\Phi(\boldsymbol{u}_{n+1})\| < \delta$. If yes, stop. Otherwise increment n and repeat from 2.

This procedure allows us to carry out geodesic projection on $P^*(\mathcal{N}_m)$.

4.6.7 Results

In this section, we briefly consider application of the numerical projection method developed in 4.6.6 to the model reduction of of multivariate Gaussian distributions in $P^*(\mathcal{N}_n)$ which have nonzero means $\boldsymbol{\mu}$ and diagonal covariance matrices $\boldsymbol{\Sigma} = \text{diag}(\sigma_1, \ldots, \sigma_n)$. Given a distribution defined by $[\boldsymbol{\mu}, \boldsymbol{\Sigma}]$ and supposing that the first r variances $\sigma_1, \ldots, \sigma_r$ are much larger than the remaining variances $\sigma_{r+1}, \ldots, \sigma_n$, we can reduce the dimension by projecting onto the submanifold defined by $\sigma_{r+1} = \varepsilon, \ldots, \sigma_n = \varepsilon$. Carrying out the method 4.6.6 effectively yields the same point that would be obtained through flat space projection given by $[\boldsymbol{\mu}, \text{diag}(\sigma_1, \ldots, \sigma_r, \varepsilon, \ldots, \varepsilon)]$ where the subdominant variances have simply been replaced by ε. This negative result shows that the principal component submanifolds considered in sections 4.6.3 and 4.6.4 do not capture the curved geometry of the full manifold of multivariate Gaussian distributions and a different approach is required to obtain useful information geometric model reduction.

CHAPTER 5: CONCLUSION

In section 1.4, the contributions of this thesis were prefaced before discussing in detail the model reduction approaches that were investigated. Having now described the acceleration procedures and results obtained from application to both deterministic and stochastic systems, we can now conclude the work presented in the previous chapters.

5.1: NONLINEAR MODEL REDUCTION

The procedure described in 2.2 was able to achieve nonlinear model reduction for both the deterministic FPU system as well as the stochastic MD simulation by compression of both the state and the mesoscale forward map of the dynamics. It was shown that for MD, the nonlinear compression of the state provided by AEs outperformed linear POD across various low-dimensional spaces indicating that nonlinear transformations indeed outperform linear transformations in extracting the dominant modes of a nonlinear dynamical sytem. Furthermroe, the AE forward map showed more accurate predictions of the state than the POD forward map for sufficiently large dimensions suggesting that nonlinear compression allows for more robust representation of the slow modulation inherent in the dynamics. Not only were the forward maps an accurate forward integrator of the system at the mesocale over training data, it was able to extrapolate the overall protein configuration beyond the simulation data for up to three mesosteps at significantly reduced computational cost as each mesostep consisted of $6 \cdot 10^5$ microsteps. This is a significant acceleration in comparison to DNS.

5.1.1 Molecular Dynamics acceleration procedure

Based on the positive results described above, the procedure detailed in 2.2 could provide robust acceleration of DNS through a computational procedure based on the following sequence of stages. First, DNS of the system is carried out at high computational cost and data is extracted to simultaneously build a mesoscale forward map extrapolation procedure based on DNNs. The extrapolation procedure is then

periodically invoked to advance the system at the mesoscale reduced computational cost for a limited number of mesosteps. This stage predicts the slow modulation of the system over the mesoscale, ignoring microscale dynamics. These positions can be further adjusted to conform to physical constraints of the problem in a corrector stage where the small scales of the problem are equilibrated. This procedure then repeats and benefits from prediction of the slow modulation of the system. The computational framework for the above MD acceleration algorithm was developed as a part of the contributions of this thesis. It automatically invokes the external MD code and periodically extracts the data to generate DNNs and advance the MD state through application of the learned mesoscale forward map. Extensive testing and validation of such a procedure is a promising future direction of the contributions presented in this thesis.

5.2: STOCHASTIC MODEL REDUCTION

5.2.1 Extrapolation of moments approach

Stochastic model reduction of MD was investigated in section 3.2 and was intended to model the inherent randomness of the system by considering stochastic processes as potential representations of dominant modes. It was verified that consideration of the weights as stochastic processes is justified as the weights exhibit the behavior of a continuous stochastic process. While tangible acceleration of MD was not achieved, it was verified that the procedure could be used to accelerate SGD convergence validating that extrapolation of the stochastic processes defined by the DNN weights was consistent approach.

5.2.2 Information geometric framework

The theoretical background for stochastic model reduction using an information geometric approach to geodesic projection onto submanifolds was developed in chapter 4. The main result was to construct a numerical method for geodesic projection onto the principal component submanifold $P^*(\mathcal{N}_m)$ of multivariate Gaussian distributions with nonzero means and diagonal covariance. Ultimately, this procedure failed to provide better results than flat-space projection which is equivalent to truncation of the Karhunen-Loève expansion. This negative result shows that to gain significant insight from information geometric projection, a non-diagonal covariance matrix must be considered which leads to complex curved geometries. This motivates the future research directions discussed below based on theoretical tools to effectively flatten the manifold $S(\mathcal{N}_m)$ of Gaussian distributions so that projection reduces to the standard linear form.

5.3: FUTURE WORK

Recall that model reduction of a Gaussian stochastic process $w_t(\omega)$ proceeds by first estimating its joint distribution

$$p(w_n, t_n; \ldots; w_1, t_1) = (2\pi)^{-n/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^T \boldsymbol{U}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^T(\boldsymbol{w}-\boldsymbol{\mu})\right)$$

from SVD of a data matrix. This joint distribution can be viewed as a point in the nonlinear manifold $S(\mathcal{N}_m)$ of multivariate Gaussian distributions. The information geometric approach to model reduction is to project this point onto a lower-dimensional submanifold corresponding to truncation of $p(w_n, t_n; \ldots; w_1, t_1)$ to $m < n$ dominant modes defined by the m largest variances.

Projection in $S(\mathcal{N}_m)$ with standard coordinates $\boldsymbol{\xi} = [\boldsymbol{\mu}, \boldsymbol{\Sigma}]$ becomes an intractable problem because of the complex, curved geometry induced by the the Fisher metric in these coordinates. A promising alternative approach to this problem can be pursued based on the theory of dually flat spaces developed extensively in [6]. In this work, a nonlinear coordinate transformation is developed for exponential families of probability densities

$$p(\mathbf{x}, \boldsymbol{\xi}) = \exp\left[C(\mathbf{x}) + \sum_{i=1}^{n} \xi^i F_i(\mathbf{x}) - \psi(\xi)\right]$$

which includes multivariate Gaussian distributions. This coordinate transformation causes the Christoffel coefficients Γ^i_{jk} to vanish over the manifold rendering the geometry flat and thus geometric computations to the linear case. This motivates research into a procedure with the following structure. First, a statistical problem is formulated in standard coordinates and transformed into the flat space representation using the tools developed in [6]. The flat space problem is solved with standard approaches to Euclidean spaces. Finally, the inverse transform can be used to map the problem back to standard coordinates creating a workflow similar to that of working with the Fourier transform.

APPENDIX A: DIFFERENTIAL GEOMETRY

A.1: ELEMENTARY DIFFERENTIAL GEOMETRY

Information geometry is concerned with a particular class of manifolds whose geometry is derived from families of probability distributions. Functionals from the information theory of random processes provide a useful notion of distance between pairs of probability distributions from these families and forms the concept of a statistical manifold whose geometry can inform statistical relationships. Hence, before defining the concept of a statistical manifold, it is necessary to introduce the basic notions of differential geometry

Differential geometry is concerned with the properties of curves, surfaces and the higher-dimensional analogues of these well-known geometric objects. All such objects can be defined as a particular manifold, the central object of study in differential geometry. Manifolds can be viewed both in a geometrically concrete manner as a surface embedded in a higher dimensional space, or from a more abstract, intrinstic point of view without reference to such an external space. Intuitively speaking, an n-dimensional real manifold is a space whose local structure around an arbitrary point is isomorphic to an open subset of $\mathbb{R}^n$. The isomorphism in this context is meant to preserve the local differentiable structure and so is naturally taken as a diffeomorphism, a smooth map with a smooth inverse. The usefulness of this local property stems from the idea that manifolds provide a class of spaces that, while considerably more general than finite dimensional vector spaces, can still be understood by patching together these local descriptions which reduce to an understanding of $\mathbb{R}^n$.

Many concepts from euclidean geometry, the study of flat vector spaces $\mathbb{R}^n$, have useful generalizations to manifolds. Often these notions begin from the local description of a manifold given by the tangent space, a vector space associated to a particular point which intuitively represents the local linearization around that point. The global distance defined on a vector space then becomes a local distance on each tangent space comprising the metric of the manifold which varies pointwise. The notion of a straight line becomes a curve that is locally straight and minimizes the metric-induced distance between two points on the manifold. The global concept of a coordinate system instead becomes a collection of local coordinate charts which obey consistency conditions on overlapping patches. The transfer of these concepts to manifolds allows for many of the same vector space operations to be carried out in a curved space.

A.1.1 Topological manifolds

Manifolds are characterized by the equivalence of their local neighborhoods to $\mathbb{R}^n$. Hence, it is natural to begin with the definition of a topological manifold in terms of the open sets of a topological space and then augment this construction with additional smooth structure to obtain a $\mathcal{C}^\infty$ manifold.

Definition A.1. A topology on a set M is a set $\mathcal{O} \subseteq \mathcal{P}(M)$ such that:

i. The empty set $\varnothing \in \mathcal{O}$ and $M \in \mathcal{O}$,

ii. If $U \in \mathcal{O}$ and $V \in \mathcal{O}$ then $U \cap V \in \mathcal{O}$,

iii. If $U_\alpha \in \mathcal{O}, \alpha \in I$ then $\bigcup_{\alpha \in I} U_\alpha \in \mathcal{O}$.

A topology represents weakest structure we can define on a set of points that allows for the important notions of convergence of a sequence to a limit and continuity of maps between sets. There are two additional properties a topological space can have which are important for building a smooth manifold structure over the topology. These are the notions of Hausdorff and paracompactness.

Definition A.2. A topological space $(M, \mathcal{O})$ is paracompact if every open cover has a locally finite open refinement.

The property means that points can be separated by the open sets of the topological space and implies the uniqueness of important objects such as limits of sequences.

Definition A.3. A topological space $(M, \mathcal{O})$ is said to be Hausdorff if, for any two distinct points, there exist non-intersecting open neighborhoods of these two points.

Topological spaces which are both Hausdorff and paracompact admit partitions of unity, a family of continuous maps with properties that allow for the extension of local geometric objects to global objects. With these concepts in mind, we can now state the definition of a topological manifold.

Definition A.4. A Hausdorff, second-countable, paracompact topological space $(M, \mathcal{O})$ is called an n-dimensional topological manifold if for every $p \in M$ there exists a neighborhood U and a homeomorphism ϕ from U onto an open set $\phi(U) \subseteq \mathbb{R}^n$. The pair (U, ϕ) is called a chart with coordinate map ϕ. The chart (U, ϕ) is centered at $p \in U$ if $\phi(p) = 0$.

A.1.2 Smooth manifolds

Two charts $(U, \phi : U \to \mathbb{R}^n), (V, \psi : V \to \mathbb{R}^n)$ of a topological manifold are compatible in the sense that both of the transition maps

$$\phi \circ \psi^{-1} : \psi(U \cap V) \to \phi(U \cap V)$$

$$\psi \circ \phi^{-1} : \phi(U \cap V) \to \psi(U \cap V)$$

are homeomorphisms on their respective domains. Hence, changing coordinates preserves the topological structure of the manifold. These charts provide a local coordinate system for the manifold defined on a particular open set. We can build a global coordinate system by forming a collection of compatible charts whose neighborhoods together form a cover for the entire space. Note that chart compatibility for topological manifolds amounted to transition functions being elements of $\mathcal{C}^0$ which extends naturally to a notion of $\mathcal{C}^k$-compatibility and the concept of a $\mathcal{C}^k$ atlas.

Definition A.5. A $\mathcal{C}^k$ *atlas* on a locally Euclidean space M is a collection $\mathfrak{U} = \{(U_\alpha, \phi_\alpha)\}$ of pairwise $\mathcal{C}^k$-compatible charts that cover M, i.e., such that $M = \bigcup_\alpha U_\alpha$.

An atlas $\mathcal{M}$ on a locally Euclidean space is maximal if it is not contained in a larger atlas. A smooth or $\mathcal{C}^\infty$ manifold is a topological manifold M together with a maximal $\mathcal{C}^\infty$ atlas $\mathcal{A}$ and is denoted as the triple $(M, \mathcal{O}, \mathcal{A})$. Often only the underlying space M is explicitly written with the additional structure assumed. A manifold is said to have dimension n if all of its connected components have dimension n. It is sometimes useful to think of coordinates componentwise in the following sense. Let $r^1, \ldots, r^n$ denote the standard coordinates on $\mathbb{R}^n$. If $(U, \phi : U \to \mathbb{R}^n)$ is a chart of a manifold, let $x^i = r^i \circ \phi$ be the ith component of ϕ and write $\phi = (x^1, \ldots, x^n)$ and $(U, \phi) = (U, x^1, \ldots, x^n)$. So for $p \in U$, $(x^1(p), \ldots, x^n(p)) \in \mathbb{R}^n$. The functions $x^1, \ldots, x^n$ are called coordinates on U.

An important example of a smooth manifold is the product manifold between two existing manifolds. If M and N are C^∞ manifolds, then $M \times N$ with its product topology is Hausdorff and second countable. To construct an atlas on the product space, define the product of two set maps $f : X \to X'$, $g : Y \to Y'$ is

$$f \times g : X \times Y \to X' \times Y', (f \times g)(x, y) = (f(x), g(y))$$

If $\{(U_\alpha, \phi_\alpha)\}$ and $\{(V_i, \psi_i)\}$ are C^∞ atlases for the manifolds M and N of dimensions m and n, respectively, then the collection

$$\{(U_\alpha \times V_\beta, \phi_\alpha \times \psi_\beta : U_\alpha \times V_\beta \to \mathbb{R}^m \times \mathbb{R}^n)\}$$

of charts is a C^∞ atlas on $M \times N$. Thus, $M \times N$ is a C^∞ manifold of dimension $m + n$.

A.1.3 Functions on manifolds

We would like to transfer familiar notions such as smooth maps and partial derivatives of maps to manifolds. The manifold itself is not a vector space which eliminates the possibility of defining these notions directly on the manifold. Instead, we use coordinate charts to pull the notion back to subsets $\mathbb{R}^n$ where the concept can be defined as usual. Using the $\mathcal{C}^\infty$-compatibility of charts in a $\mathcal{C}^\infty$ atlas, the smoothness of a map is easily seen to be independent of the choice of charts and is thus a well defined concept.

Let M be a smooth manifold of dimension n. A function $f : M \to \mathbb{R}$ is said to be $\mathcal{C}^\infty$ at $p \in M$ if there is a chart (U, ϕ) about p in M such that $f \circ \phi^{-1} : \phi(U) \to \mathbb{R}$, is C^∞ at $\phi(p)$. The function f is $\mathcal{C}^\infty$ on M if it is $\mathcal{C}^\infty$ at every point of M. This definition is independent of the chart (U, ϕ) for if $f \circ \phi^{-1}$ is C^∞ at $\phi(p)$ and (V, ψ) is any other chart about p in M, then on $\psi(U \cap V)$

$$f \circ \psi^{-1} = (f \circ \phi^{-1}) \circ (\phi \circ \psi^{-1})$$

which is $\mathcal{C}^\infty$ at $\psi(p)$. The concept of smooth maps between manifolds is defined analogously. Let N and M be manifolds of dimension n, m. A continuous map $F : N \to M$ is $\mathcal{C}^\infty$ at a point p in N if there are charts (V, ψ) about $F(p)$ in M and (U, p) about $p \in N$ such that the composition $\psi \circ F \circ \phi^{-1}$, a map from the open subset $\phi(F^{-1}(V) \cap U)$ of $\mathbb{R}^n$ to $\mathbb{R}^m$, is C^∞ at $\phi(p)$. The map $F : N \to M$ is said to be C^∞ if it is C^∞ at every point of N.

Finally, we transfer the notion of partial derivatives from Eucliean space to a coordinate chart on a manifold.Let M be a manifold of dimension n with (U, ϕ) a chart and f a C^∞ function. The function ϕ has n components $x^1, \ldots, x^n$ so that if $r^1, \ldots, r^n$ are the standard coordinates on $\mathbb{R}^n$, then $x^i = r^i \circ \phi$. For $p \in U$, we define the partial derivative $\partial f / \partial x^i$ of f with respect to x^i at p to be

$$\frac{\partial}{\partial x^i}|_p f := \frac{\partial f}{\partial x^i}(p) := \frac{\partial (f \circ \phi^{-1})}{\partial r^i}(\phi(p)) = \frac{\partial}{\partial r^i}|_{\phi(p)}(f \circ \phi^{-1})$$

This definition of the derivative will yield a differential operator that plays the crucial role of a basis for the tangent vector space at a point.

An important example of a smooth map is the inclusion map $i_{q_0} : M \to M \times N$, $i_{q_0}(p) = (p, q_0)$ immersing M into the product manifold $M \times N$ as a submanifold. The smoothess can be seen by considering $p \in M$ be arbitrary and letting (U, ϕ) be a coordinate neighborhood about p. Let (V, ψ) be a coordinate neighborhood of $q_0 \in N$. Then $(U \times V, \phi \times \psi)$ is a coordinate neighborhood of $(p, q_0) = i_{q_0}(p)$. Consider

$$(\phi \times \psi) \circ i_{q_0} \circ \phi^{-1} : \phi(U \cap i_{q_0}^{-1}(U \times V)) \subset \mathbb{R}^m \to \mathbb{R}^{m+n}$$

and let $\phi(p) = (a^1, \ldots, a^m)$.

$$\begin{aligned} (\phi \times \psi) \circ i_{q_0} \circ \phi^{-1}(a^1, \ldots, a^m) &= (\phi \times \psi)(p, q_0) \\ &= (\phi(p), \psi(q_0)) \end{aligned}$$

which is C^∞ as ϕ is C^∞ at p and ψ is C^∞ at q_0.

A.1.4 Tangent space

The simplest and most intrinsic way to define a tangent vector at a point p in a manifold M is through the idea of a differential operator acting on a function at p. The differential operator can be imbued with a direction by defining it as the derivative of a function along some smooth curve $\gamma : \mathbb{R} \to M$. Hence, given a smooth curve γ such that $\gamma(0) = p$ the directional derivative operator at p along γ is the linear map

$$\begin{aligned} X_{\gamma,p} : \mathcal{C}^\infty &\to \mathbb{R} \\ f &\mapsto (f \circ \gamma)'(0) \end{aligned}$$

The tangent space can then be defined as the vector space over $\mathbb{R}$ with the underlying set

$$T_pM := \{X_{\gamma,p} | \gamma \text{is a smooth curve through } p\}$$

and addition and multiplication operations

$$\begin{aligned} \oplus : T_pM \times T_pM &\rightarrow T_pM \\ (X_{\gamma,p}, X_{\delta,p}) &\mapsto X_{\gamma,p} \oplus X_{\delta,p} \end{aligned}$$

$$\begin{aligned} \odot : \mathbb{R} \times T_pM &\rightarrow T_pM \\ (\lambda, X_{\gamma,p}) &\mapsto \lambda \odot X_{\gamma,p} \end{aligned}$$

that can be defined to function consistently in terms of directional derivatives along curves. We can find a basis for T_pM in terms of a chart $(U, x^1, \ldots, x^n)$ about p by observing the action of a directional derivative operator on a function $f \in \mathcal{C}^\infty(M)$

$$\begin{aligned} X_{\gamma,p}(f) &= (f \circ \gamma)'(0) \\ &= \frac{\mathrm{d}}{\mathrm{d}t}(f \circ \phi^{-1} \circ \phi \circ \gamma)(0) \\ &= \frac{\partial}{\partial r^i}(f \circ \phi^{-1})(\phi(p))(x^i \circ \gamma)'(0) \\ &= \left[\sum_{i=1}^{n} \dot{\gamma}^i(0) \left(\frac{\partial}{\partial x^i}\right)_p\right] f \end{aligned}$$

where $\left(\frac{\partial}{\partial x^i}\right)_p$is defined as the operator

$$\left(\frac{\partial}{\partial x^i}\right)_p f := \frac{\partial(f \circ \phi^{-1})}{\partial r^i}(\phi(p))$$

Hence the $\left(\frac{\partial}{\partial x^i}\right)_p$ span T_pM and linear independence can be shown similarly. The tangent space can be written in terms of this basis

$$T_pM = \left\{ c^i \left(\frac{\partial}{\partial x^i}\right)_p \middle| (c^1, \ldots, c^n) \in \mathbb{R}^n \right\}$$

Conversion between tangent vectors in coordinate systems $(U, x^1, \ldots, x^n)$ and $(V, y^1, \ldots, y^n)$ is carried out using

$$\left(\frac{\partial}{\partial y^j}\right)_p = \left(\frac{\partial x^i}{\partial y^j}\right)_p \left(\frac{\partial}{\partial x^i}\right)_p$$

$$\left(\frac{\partial}{\partial x^i}\right)_p = \left(\frac{\partial y^j}{\partial x^i}\right)_p \left(\frac{\partial}{\partial y^j}\right)_p$$

where the $\left(\frac{\partial x^i}{\partial y^j}\right)_p$ terms can be thought of the as comprising the manifold generalization of the jacobian of the transition map.

A.1.5 Differential of a map

Let $F : N \to M$ a C^∞ map between manifolds. At each $p \in N$, F induces a linear map of tangent spaces, called its differential at p,

$$F_* : T_pN \to T_{F(p)}M$$

as follows. If $X_p \in T_pN$, then $F_*(X_p)$ is the tangent vector in $T_{F(p)}M$ defined by

$$(F_*(X_p))f = X_p(f \circ F) \in \mathbb{R} \text{ for } f \in C^\infty_{F(p)}(M).$$

To see this, observe that

$$\begin{aligned}
X'(f) &= \left(\frac{\partial}{\partial \xi^i}\right)_x (f \circ \phi) \\
&= \frac{\partial \overline{(f \circ \phi)}}{\partial \xi^i} \circ \xi x \\
&= \frac{\partial (f \circ \phi \circ \xi^{-1})}{\partial \xi^i} \circ \xi(x) \\
&= \frac{\partial (f \circ \rho^{-1} \circ \rho \circ \phi \circ \xi^{-1})}{\partial \xi^i} \circ \xi(x) \\
&= \left(\left[\frac{\partial (f \circ \rho^{-1})}{\partial \rho^j}\right]_{\rho(\phi(x))} \left[\frac{\partial (\rho^j \circ \phi \circ \xi^{-1})}{\partial \xi^i}\right]\right) \circ \xi(x) \\
&= \left[\frac{\partial (f \circ \rho^{-1})}{\partial \rho^j}\right]_{\rho(\phi(x))} \left[\frac{\partial (\rho^j \circ \phi \circ \xi^{-1})}{\partial \xi^i} \circ \xi(x)\right] \\
&= \left(\frac{\partial (\rho^j \circ \phi)}{\partial \xi^i}\right)_x \left(\frac{\partial}{\partial \rho^j}\right)_{\phi(x)} f
\end{aligned}$$

which shows both provide the same action on an arbitrary $f \in \mathcal{F}(N)$.

A.1.6 Vector fields

A vector field is a function $X : p \mapsto X_p$ where $X_p \in T_p(S)$. Let ∂_i denote $\frac{\partial}{\partial \xi^i}$ and note that the mapping $\partial_i : p \mapsto (\partial_i)_p$ forms a vector field for each i. Any vector field can be written in terms of its coordinate

functions X^i as $X = X^i \partial_i$. We can convert between coordinate systems $[\xi^i]$ and $[\rho^j]$ via

$$\tilde{X}^j = X^i \frac{\partial \rho^j}{\partial \xi^i}, X^i = \tilde{X}^j \frac{\partial \xi^i}{\partial \rho^j}$$

Denote the family of vector fields on S by $\mathcal{T}(S)$ or $\mathcal{T}$. Note that $\mathcal{T}$ is closed under linear combinations and hence is a linear space.

A.1.7 Submanifolds

Let M, S be manifolds with $M \subset S$. If $[\xi^1, \ldots, \xi^n]$ coordinates for S and $[u^1, \ldots, u^m]$ coordinates for M then M is a submanifold of S if

i. $\xi^i|_M$ is a C^∞ function on M for each i.

ii. Let $B_a^i = \left(\frac{\partial \xi^i}{\partial u^a}\right)_p$ and $B_a = [B_a^1, \ldots, B_a^n] \in \mathbb{R}^n$. Then for each $p \in M$, $\{B_1, \ldots, B_m\}$ are linearly independent.

iii. For any open subset $W \subset M$, there exists U an open subset of S such that $W = M \cap U$.

We can define a natural example of a submanifold as follows

$$M = \{p \in S | \xi^i(p) = c^i, m+1 \leqslant i \leqslant n\}$$

Furthermore, if M is an m-dimensional submanifold of S with coordinates $[u^a]$ and $\{c^{m+1}, \ldots, c^n\}$ are $n - m$ real numbers then there exists $U \subset S$ and coordinates $[\xi^i]$ such that

$$M \cap U = \{p \in U | \xi^i(p) = c^i, m+1 \leqslant i \leqslant n\}$$

and $u^a|_{M \cap U} = \xi^a | M \cap U$.

A.1.8 Riemannian metrics

Let M be a manifold on which we define a field of inner products $\langle , \rangle_p$ at each $p \in M$ defined on $T_p(M)$. This forms a tensor field of rank $(0, 2)$ and is called a *Riemannian metric*. A metric, typically denoted g, is independent from structure of M hence we can consider infinitely many of them.

Given coordinates $[\xi^i]$ for M and vectors $D = D^i(\partial_i)_p, D' = D'^i(\partial_i)_p$ the metric operates component wise as

$$\langle D, D' \rangle_p = g_{ij}(p) D^i D'^j$$

where $g_{ij}(p) = \langle (\partial_i)_p, (\partial_j)_p \rangle$ are the components of g at p with respect to coordinates $[\xi^i]$.

Given another coordinate system $[\rho^j]$ on M we have the following transformation laws for g

$$\tilde{g}_{kl} = g_{ij} \left(\frac{\partial \xi^i}{\partial \rho^k} \right) \left(\frac{\partial \xi^j}{\partial \rho^l} \right) \qquad\qquad g_{ij} = \tilde{g}_{kl} \left(\frac{\partial \rho^k}{\partial \xi^i} \right) \left(\frac{\partial \rho^l}{\partial \xi^j} \right)$$

where $\tilde{g}_{kl} = \langle \tilde{\partial}_k, \tilde{\partial}_l \rangle$, $\tilde{\partial}_k \equiv \frac{\partial}{\partial \rho^k}$.

A.1.8.1 Induced metric on a submanifold

Suppose we have a submanifold $S \subset M$ embedded via a map $\phi : S \to M$ which could simply be the identity (inclusion) map. If M has a metric g, this induces a metric $(g|_S)$ on S via

$$(g|_S)(X, Y) = g(d\phi(X), d\phi(Y))$$

i.e., the pullback of g by ϕ, sometimes written $\phi^* g$. Let $[\xi^i]$ be coordinates on S and $[x^j]$ coordinates on M. To derive the resulting component expressions for this induced metric, we compute

$$\begin{aligned}
(g|_S)_{ij} &= g|_S \left(\left(\frac{\partial}{\partial \xi^i} \right)_p, \left(\frac{\partial}{\partial \xi^j} \right)_p \right) \\
&= \phi^* g \left(\left(\frac{\partial}{\partial \xi^i} \right)_p, \left(\frac{\partial}{\partial \xi^j} \right)_p \right) \\
&= g \left(\phi_* \left(\frac{\partial}{\partial \xi^i} \right)_p, \phi_* \left(\frac{\partial}{\partial \xi^j} \right)_p \right) \\
&= g \left(\left(\frac{\partial (x^l \circ \phi)}{\partial \xi^i} \right)_p \left(\frac{\partial}{\partial x^l} \right)_{\phi(p)}, \left(\frac{\partial (x^k \circ \phi)}{\partial \xi^j} \right)_p \left(\frac{\partial}{\partial x^k} \right)_{\phi(p)} \right) \\
&= \left(\frac{\partial (x^l \circ \phi)}{\partial \xi^i} \right) \left(\frac{\partial (x^k \circ \phi)}{\partial \xi^j} \right)_p g \left(\left(\frac{\partial}{\partial x^l} \right)_{\phi(p)}, \left(\frac{\partial}{\partial x^k} \right)_{\phi(p)} \right) \\
&= \left[\frac{\partial (x^l \circ \phi \circ \xi^{-1})}{\partial \xi^i} \circ \xi(p) \right] \left[\frac{\partial (x^k \circ \phi \circ \xi^{-1})}{\partial \xi^j} \circ \xi(p) \right] g_{lk}(\phi(p)) \\
&= \left(\frac{\partial x^l}{\partial \xi^i} \right)_{\xi(p)} \left(\frac{\partial x^k}{\partial \xi^j} \right)_{\xi(p)} g_{lk}(\phi(p))
\end{aligned}$$

A.1.9 Affine connections and covariant derivatives

Let M be a d-dimensional smooth manifold. Another useful structure we can augment M with is an affine connection which provides an association between different tangent spaces. It can be thought of as defining a notion of prarallel transport. We require this mapping to reduce to a linear transformation when the tangent spaces are infinitesimally close, informally speaking. Hence, if we denote transport from $p \in M$ to $p' \in M$ by $\prod_{p,p'}((\partial_j)_p)$ then the connection takes the form

$$\prod_{p,p'}((\partial_j)_p) = (\partial_j)_{p'} - \mathrm{d}\xi^i(\Gamma^k_{ij})_p(\partial_k)_{p'}$$

in the infinitesimal case where $(\Gamma^k_{ij})_p$ are d^3 coefficients known as the connection coefficients. This differential operator extends to an ODE defining parallel transport along a curve $\gamma(t)$ of finite length

$$\dot{X}^k(t) + \dot{\gamma}^i(t)X^j(t)(\Gamma^k_{ij})_{\gamma(t)} = 0$$

This gives rise to the equivalent notion of a covariant derivative of one vector field by another. Hence, if $X, Y \in \mathcal{V}(M)$ then

$$\nabla_X Y = X^i\{\partial_i Y^k + Y^j\Gamma^k_{ij}\}\partial_k$$

and can also be used to define the geodesic equations

$$\ddot{\gamma}^k(t) + (\Gamma^k_{ij})_{\gamma(t)}\dot{\gamma}^i(t)\dot{\gamma}^j(t) = 0$$

for the coordinate components $\gamma^i = x^i \circ \gamma$ of a curve $\gamma(t) : \mathbb{R} \to M$ that generalizes a straight line in the sense of parallel transporting its tangent vector along itself.

BIBLIOGRAPHY

[1] P. A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematicae*, 80(2):199–220, Jan. 2004.

[2] R. L. Akkermans and W. J. Briels. A structure-based coarse-grained model for polymer melts. *The Journal of Chemical Physics*, 114(2):1020–1031, 2001.

[3] S. AMARI. DIFFERENTIAL GEOMETRY OF CURVED EXPONENTIAL-FAMILIES - CURVATURES AND INFORMATION LOSS. *ANNALS OF STATISTICS*, 10(2):357–385, 1982.

[4] S.-I. Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[5] S.-i. Amari, K. Kurata, and H. Nagaoka. Information geometry of Boltzmann machines. *IEEE Transactions on neural networks*, 3(2):260–271, 1992.

[6] S.-i. Amari and H. Nagaoka. *Methods of Information Geometry*, volume 191. American Mathematical Soc., 2007.

[7] D. Amsallem, J. Cortial, K. Carlberg, and C. Farhat. A method for interpolating on manifolds structural dynamics reduced-order models. *International journal for numerical methods in engineering*, 80(9):1241–1258, 2009.

[8] D. Amsallem and C. Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.

[9] D. Amsallem and C. Farhat. An online method for interpolating linear parametric reduced-order models. *SIAM Journal on Scientific Computing*, 33(5):2169–2198, 2011.

[10] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.

[11] A. C. Antoulas, C. A. Beattie, and S. Gugercin. Interpolatory model reduction of large-scale dynamical systems. In *Efficient Modeling and Control of Large-Scale Systems*, pages 3–58. Springer, 2010.

[12] Z. Artstein. On singularly perturbed ordinary differential equations with measure-valued limits. *Proceedings of Equadiff 10*, pages 15–26, 2002.

[13] P. Astrid, S. Weiland, K. Willcox, and T. Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008.

[14] Z. Bai. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied numerical mathematics*, 43(1-2):9–44, 2002.

[15] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation'method: Application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[16] U. Baur, C. Beattie, P. Benner, and S. Gugercin. Interpolatory Projection Methods for Parameterized Model Reduction. *Siam Journal on Scientific Computing*, 33(5):2489–2518, 2011.

[17] U. Baur and P. Benner. Model reduction for parametric systems using balanced truncation and interpolation. *at-Automatisierungstechnik*, 57(8):411–419, 2009.

[18] U. Baur, P. Benner, A. Greiner, J. G. Korvink, J. Lienemann, and C. Moosmann. Parameter preserving model order reduction for MEMS applications. *Mathematical and Computer Modelling of Dynamical Systems*, 17(4):297–317, 2011.

[19] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, June 2003.

[20] P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, editors. *Model Reduction and Approximation: Theory and Algorithms*. Number 15 in Computational Science and Engineering. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, 2017.

[21] P. Benner and L. Feng. A robust algorithm for parametric model order reduction based on implicit moment matching. In *Reduced Order Methods for Modeling and Computational Reduction*, pages 159–185. Springer, 2014.

[22] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.

[23] G. P. Berman and F. M. Izrailev. The Fermi–Pasta–Ulam problem: Fifty years of progress. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 15(1):015104, Mar. 2005.

[24] A. Bietti and J. Mairal. Invariance and Stability of Deep Convolutional Representations. page 11.

[25] A. Bietti and J. Mairal. Group Invariance, Stability to Deformations, and Complexity of Deep Convolutional Representations. *arXiv:1706.03078 [cs, stat]*, Oct. 2018.

[26] T. Bui-Thanh, K. Willcox, and O. Ghattas. Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications. *AIAA journal*, 46(10):2520–2529, 2008.

[27] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem. The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, 242:623–647, 2013.

[28] J. Carr. *Applications of Centre Manifold Theory*, volume 35. Springer Science & Business Media, 2012.

[29] D. A. Case, T. E. Cheatham, T. Darden, H. Gohlke, R. Luo, K. M. Merz, A. Onufriev, C. Simmerling, B. Wang, and R. J. Woods. The Amber biomolecular simulation programs. *Journal of computational chemistry*, 26(16):1668–1688, 2005.

[30] S. Chaturantabut and D. C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[31] W. Chen, Q. Wang, J. Hesthaven, and C. Zhang. Physics-informed machine learning for reduced-order modeling of nonlinear problems. *Preprint*, 2020.

[32] D. T. Clarke, A. J. Doig, B. J. Stapley, and G. R. Jones. The α-helix folds on the millisecond time scale. *Proceedings of the National Academy of Sciences*, 96(13):7232–7237, 1999.

[33] P. Constantin, C. Foias, B. Nicolaenko, and R. Temam. *Integral Manifolds and Inertial Manifolds for Dissipative Partial Differential Equations*, volume 70. Springer Science & Business Media, 2012.

[34] D. De Sancho and R. B. Best. What is the time scale for α-helix nucleation? *Journal of the American Chemical Society*, 133(17):6809–6816, 2011.

[35] R. Dengler. Another derivation of generalized Langevin equations. *arXiv preprint arXiv:1506.02650*, 2015.

[36] P. Deuflhard, W. Huisinga, A. Fischer, and C. Schütte. Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Linear Algebra and its Applications*, 315(1-3):39–59, 2000.

[37] M. Drohmann, B. Haasdonk, and M. Ohlberger. Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation. *SIAM Journal on Scientific Computing*, 34(2):A937–A969, 2012.

[38] R. Everson and L. Sirovich. Karhunen–Loeve procedure for gappy data. *JOSA A*, 12(8):1657–1664, 1995.

[39] R. W. Freund. Model reduction methods based on Krylov subspaces. *Acta Numerica*, 12:267–319, 2003.

[40] R. Giryes, G. Sapiro, and A. M. Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64(13):3444–3457, 2016.

[41] D. Givon, R. Kupferman, and A. Stuart. Extracting macroscopic dynamics: Model problems and algorithms. *Nonlinearity*, 17(6):R55, 2004.

[42] A. N. Gorban and I. V. Karlin. Method of invariant manifolds and regularization of acoustic spectra. *Transport Theory and Statistical Physics*, 23(5):559–632, 1994.

[43] A. N. Gorban and I. V. Karlin. *Invariant Manifolds for Physical and Chemical Kinetics*, volume 660. Springer Science & Business Media, 2005.

[44] M. A. Grepl and A. T. Patera. A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 39(1):157–181, 2005.

[45] P. Gunupudi, R. Khazaka, and M. Nakhla. Analysis of transmission line circuits using multidimensional model reduction techniques. *IEEE Transactions on Advanced Packaging*, 25(2):174–180, 2002.

[46] C. Heil and D. Walnut. Continuous and Discrete Wavelet Transforms. *Siam Review*, 31(4):628–666, Dec. 1989.

[47] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[48] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[49] S. Izvekov, P. W. Chung, and B. M. Rice. The multiscale coarse-graining method: Assessing its accuracy and introducing density dependent coarse-grain potentials. *The Journal of Chemical Physics*, 133(6):064109, 2010.

[50] S. Izvekov and G. A. Voth. A multiscale coarse-graining method for biomolecular systems. *The Journal of Physical Chemistry B*, 109(7):2469–2473, 2005.

[51] I. Jolliffe. Principal component analysis. *Technometrics*, 45(3):276, 2003.

[52] J. N. Kani and A. H. Elsheikh. DR-RNN: A deep residual recurrent neural network for model reduction. *arXiv preprint arXiv:1709.00939*, 2017.

[53] J. Karhunen and J. Joutsensalo. Generalizations of principal component analysis, optimization problems, and neural networks. *Neural Networks*, 8(4):549–562, 1995.

[54] Y. Kifer. Random perturbations of dynamical systems. *Nonlinear Problems in Future Particle Accelerators. World Scientific*, page 189, 1988.

[55] T. Kim. Frequency-domain Karhunen-Loeve method and its application to linear dynamic systems. *AIAA journal*, 36(11):2117–2123, 1998.

[56] D. Kosambi. Statistics in function space. In *DD Kosambi*, pages 115–123. Springer, 2016.

[57] M. Kouza, C.-K. Hu, M. S. Li, and A. Kolinski. A structure-based model fails to probe the mechanical unfolding pathways of the titin I27 domain. *The Journal of chemical physics*, 139(6):08B615_1, 2013.

[58] H.-O. Kreiss. Problems with different time scales. *Acta numerica*, 1:101–139, 1992.

[59] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the Acm*, 60(6):84–90, June 2017.

[60] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[61] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[62] P. Lochak and C. Meunier. *Multiphase Averaging for Classical Systems: With Applications to Adiabatic Theorems*, volume 72. Springer Science & Business Media, 2012.

[63] M. Loeve. Probability theory: Foundations, random sequences. 1955.

[64] J. L. Lumley. The structure of inhomogeneous turbulent flows. *Atmospheric turbulence and radio wave propagation*, 1967.

[65] S. J. Marrink, H. J. Risselada, S. Yefimov, D. P. Tieleman, and A. H. De Vries. The MARTINI force field: Coarse grained model for biomolecular simulations. *The journal of physical chemistry B*, 111(27):7812–7824, 2007.

[66] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE transactions on automatic control*, 26(1):17–32, 1981.

[67] H. Mori. Transport, collective motion, and Brownian motion. *Progress of theoretical physics*, 33(3):423–455, 1965.

[68] C. Mullis and R. Roberts. Synthesis of minimum roundoff noise fixed point digital filters. *IEEE Transactions on Circuits and Systems*, 23(9):551–562, 1976.

[69] R. K. Nayak, O. B. Peersen, K. B. Hall, and A. Van Orden. Millisecond time-scale folding and unfolding of DNA hairpins using rapid-mixing stopped-flow kinetics. *Journal of the American Chemical Society*, 134(5):2453–2456, 2012.

[70] W. Noid, J.-W. Chu, G. S. Ayton, V. Krishna, S. Izvekov, G. A. Voth, A. Das, and H. C. Andersen. The multiscale coarse-graining method. I. A rigorous bridge between atomistic and coarse-grained models. *The Journal of chemical physics*, 128(24):244114, 2008.

[71] R. E. O'malley. *Singular Perturbation Methods for Ordinary Differential Equations*, volume 89. Springer, 1991.

[72] E. Paci, M. Vendruscolo, and M. Karplus. Validity of Go models: Comparison with a solvent-shielded empirical energy decomposition. *Biophysical journal*, 83(6):3032–3038, 2002.

[73] R. S. Palais. The Symmetries of Solitons. *arXiv:dg-ga/9708004*, Aug. 1997.

[74] H. Panzer, J. Mohring, R. Eid, and B. Lohmann. Parametric model order reduction by matrix interpolation. *at-Automatisierungstechnik*, 58(8):475–484, 2010.

[75] G. C. Papanicolaou. Some probabilistic problems and methods in singular perturbations. *The Rocky Mountain Journal of Mathematics*, pages 653–674, 1976.

[76] D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham III, S. DeBolt, D. Ferguson, G. Seibel, and P. Kollman. AMBER, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structural and energetic properties of molecules. *Computer Physics Communications*, 91(1-3):1–41, 1995.

[77] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kale, and K. Schulten. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry*, 26(16):1781–1802, Dec. 2005.

[78] M. Praprotnik, L. Delle Site, and K. Kremer. Adaptive resolution molecular-dynamics simulation: Changing the degrees of freedom on the fly. *The Journal of chemical physics*, 123(22):224106, 2005.

[79] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox. Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. *Physica D: Nonlinear Phenomena*, 406:132401, 2020.

[80] M. Ranzato, F. J. Huang, Y. Boureau, and Y. LeCun. Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[81] M. Rathinam and L. R. Petzold. A new look at proper orthogonal decomposition. *SIAM Journal on Numerical Analysis*, 41(5):1893–1925, 2003.

[82] W. Rawat and Z. Wang. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Computation*, 29(9):2352–2449, Sept. 2017.

[83] D. Reith, M. Pütz, and F. Müller-Plathe. Deriving effective mesoscale potentials from atomistic simulations. *Journal of computational chemistry*, 24(13):1624–1636, 2003.

[84] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[85] J. A. Sanders, F. Verhulst, and J. Murdock. *Averaging Methods in Nonlinear Dynamical Systems*, volume 59. Springer, 2007.

[86] J. M. A. Scherpen. Balancing for nonlinear systems. *Systems & Control Letters*, 21(2):143–153, 1993.

[87] S. C. Schmidler, J. E. Lucas, and T. G. Oas. Statistical estimation of statistical mechanical models: Helix-coil theory and peptide helicity prediction. *Journal of Computational Biology*, 14(10):1287–1310, 2007.

[88] B. Scholkopf, A. Smola, and K. R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, July 1998.

[89] J. M. Scholtz and R. L. Baldwin. The mechanism of alpha-helix formation by peptides. *Annual review of biophysics and biomolecular structure*, 21(1):95–118, 1992.

[90] S. Shalev-Shwartz and A. Tewari. Stochastic Methods for l(1)-regularized Loss Minimization. *Journal of Machine Learning Research*, 12:1865–1892, June 2011.

[91] M. Shensa. The Discrete Wavelet Transform - Wedding the a Trous and Mallat Algorithms. *Ieee Transactions on Signal Processing*, 40(10):2464–2482, Oct. 1992.

[92] L. Sirovich. Turbulence and the dynamics of coherent structures. I. Coherent structures. *Quarterly of applied mathematics*, 45(3):561–571, 1987.

[93] E. J. Sorin and V. S. Pande. Exploring the helix-coil transition via all-atom equilibrium ensemble simulations. *Biophysical journal*, 88(4):2472–2493, 2005.

[94] R. Swischuk, L. Mainini, B. Peherstorfer, and K. Willcox. Projection-based model reduction: Formulations for physics-based machine learning. *Computers & Fluids*, 179:704–717, 2019.

[95] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–+, Dec. 2000.

[96] K. Veroy, C. Prud'Homme, D. Rovas, and A. Patera. A posteriori error bounds for reduced-basis approximation of parametrized noncoercive and nonlinear elliptic partial differential equations. In *16th AIAA Computational Fluid Dynamics Conference*, page 3847, 2003.

[97] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, and D. A. Case. Development and testing of a general amber force field. *Journal of computational chemistry*, 25(9):1157–1174, 2004.

[98] S. Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, volume 2. Springer Science & Business Media, 2003.

[99] K. Willcox and J. Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA journal*, 40(11):2323–2330, 2002.

[100] J. Wouters and G. A. Gottwald. Stochastic model reduction for slow-fast systems with moderate time scale separation. *Multiscale Modeling & Simulation*, 17(4):1172–1188, 2019.

[101] X. Xie, C. Webster, and T. Iliescu. Closure learning for nonlinear model reduction using deep residual neural network. *Fluids*, 5(1):39, 2020.

[102] N. J. Zabusky. Fermi–Pasta–Ulam, solitons and the fabric of nonlinear and computational science: History, synergetics, and visiometrics. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 15(1):015102, Mar. 2005.

[103] R. Zimmermann. Manifold interpolation and model reduction. *arXiv preprint arXiv:1902.06502*, 2019.

[104] R. Zwanzig. Nonlinear generalized langevin equations. *Journal of Statistical Physics*, 9(3):215–220, 1973.